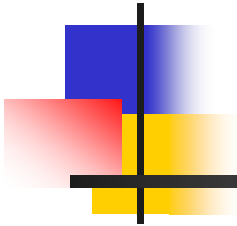


Advances in Search and Inference for Graphical Models



Rina Dechter

Donald Bren School of Computer Science
University of California, Irvine, USA

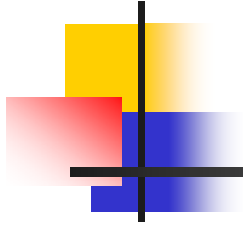
Robert Mateescu

Microsoft Research
Cambridge, UK

Radu Marinescu

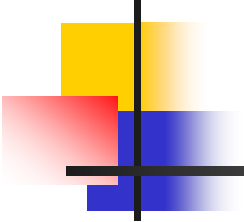
IBM
Dublin, Ireland

with contributed slides by Javier Larrosa (UPC, Spain), Simon de Givry and Thomas Schiex (INRA, France)



Outline

- Introduction
- Inference
- Search
- Compilation: AND/OR Decision Diagrams
- Software



Outline

- Introduction

- Graphical models
 - Solution Techniques

- Inference

- Search

- Compilation: AND/OR Decision Diagrams

- Software

Constraint Networks

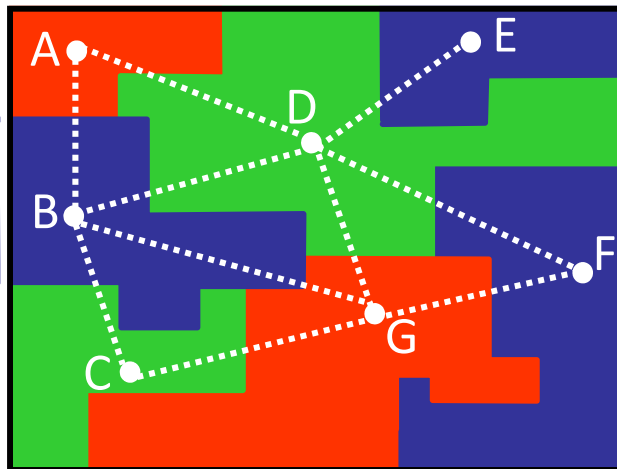
Map coloring

Variables: countries (A B C etc.)

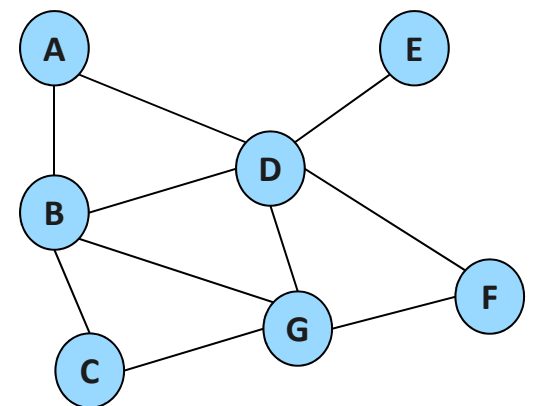
Values: colors (red green blue)

Constraints: **$A \neq B, A \neq D, D \neq E, \dots$**

A	B
red	green
red	yellow
green	red
green	yellow
yellow	green
yellow	red

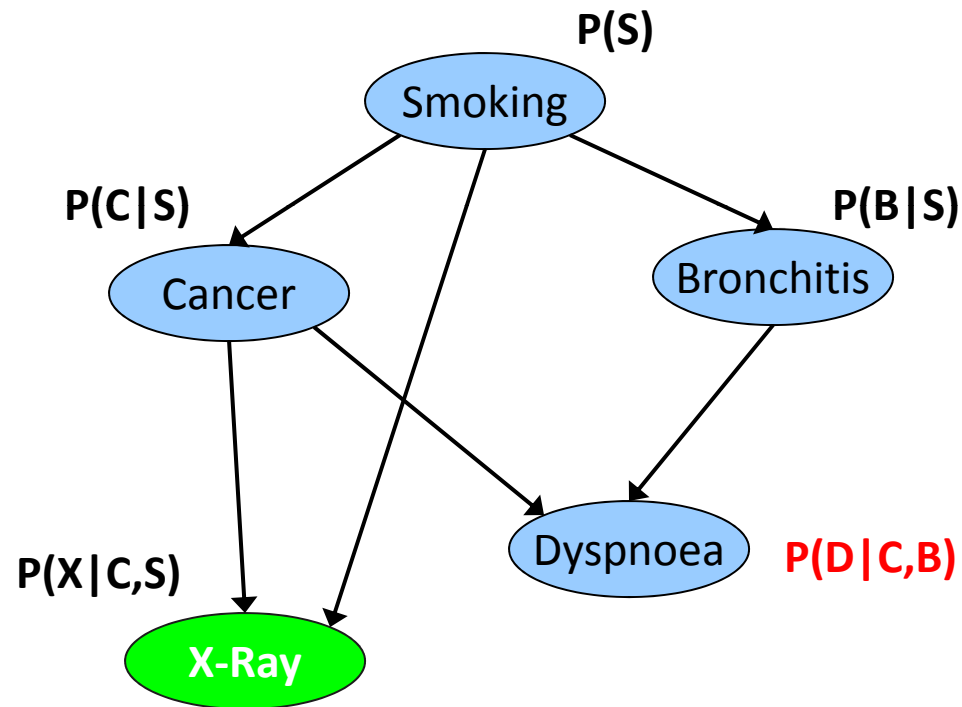


Constraint graph



Bayesian Networks

BN = (X,D,G,P)



$P(D|C,B)$

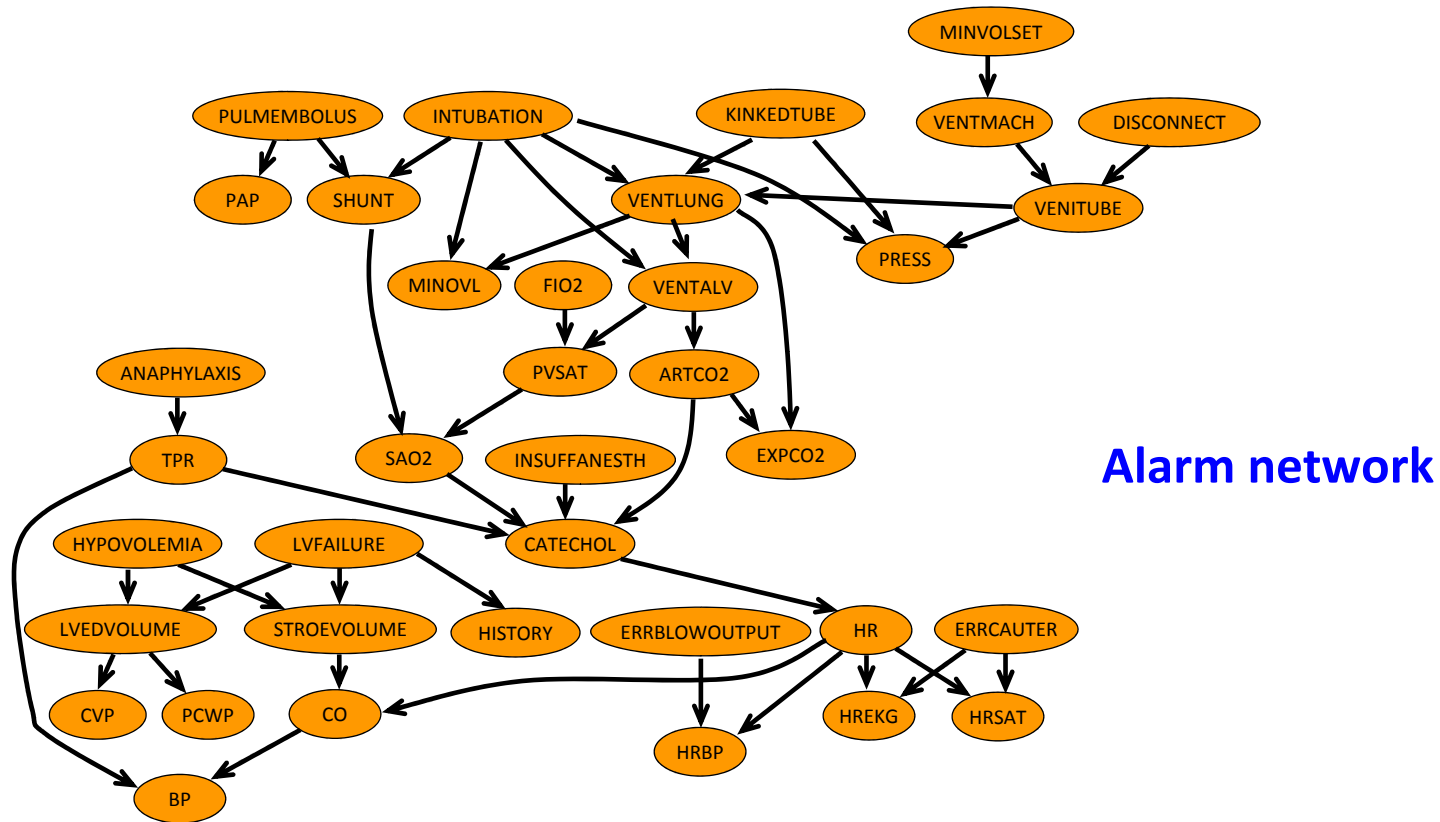
C	B	D=0	D=1
0	0	0.1	0.9
0	1	0.7	0.3
1	0	0.8	0.2
1	1	0.9	0.1

$$P(S,C,B,X,D) = P(S) \cdot P(C|S) \cdot P(B|S) \cdot P(X|C,S) \cdot P(D|C,B)$$

MPE = Find a maximum probability assignment, given evidence

MPE = find argmax $P(S) \cdot P(C|S) \cdot P(B|S) \cdot P(X|C,S) \cdot P(D|C,B)$

Monitoring Intensive-Care Patients



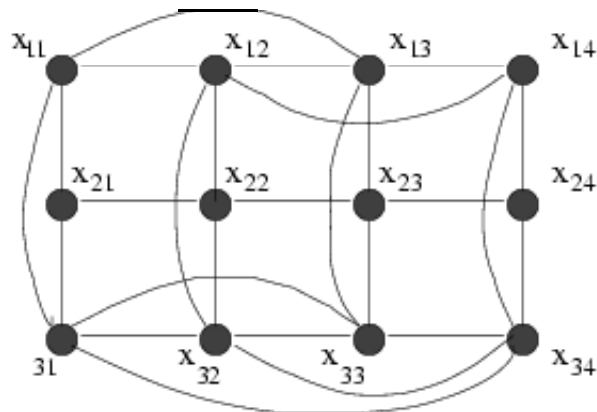
37 variables
509 parameters

<<

2³⁷

Constrained Optimization

Example: power plant scheduling



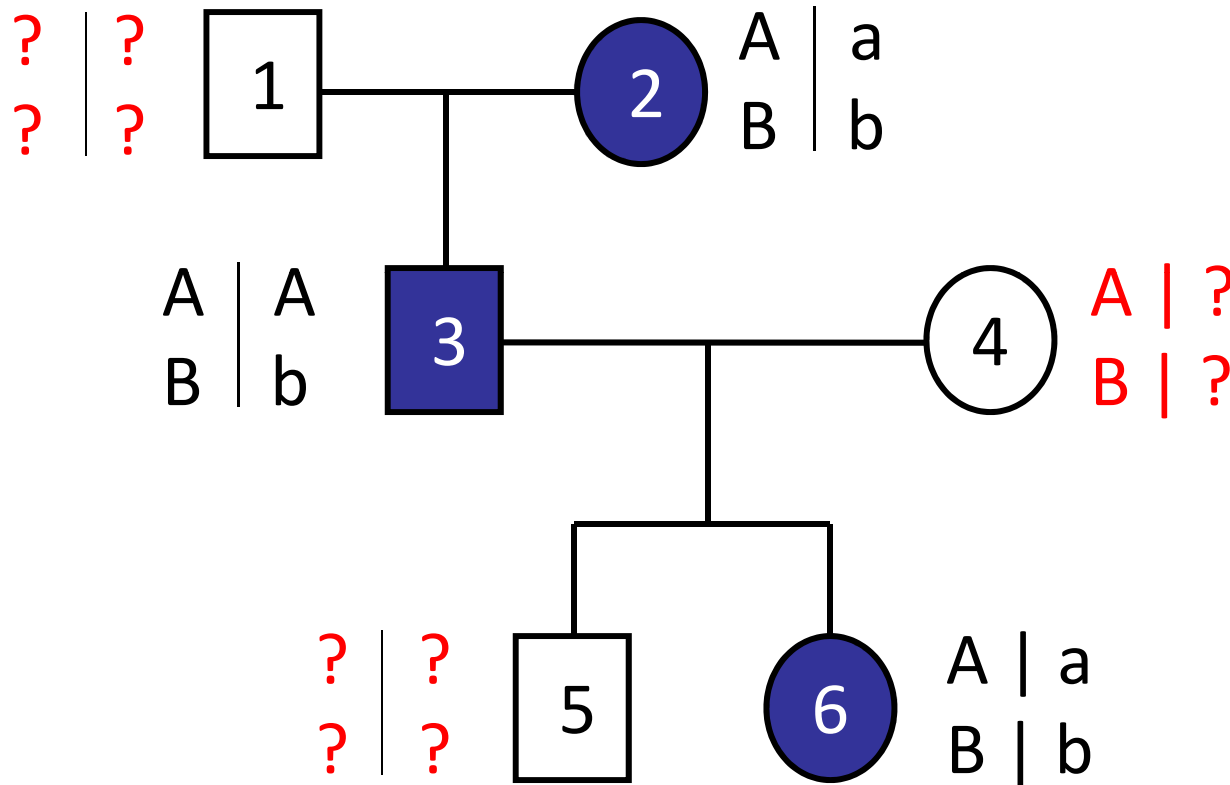
Unit #	Min Up Time	Min Down Time
1	3	2
2	2	1
3	4	1

Variables = $\{X_1, \dots, X_n\}$, domain = $\{ON, OFF\}$.

Constraints : $X_1 \vee X_2, \neg X_3 \vee X_4$, min - up and min - down time,
 power demand : $\sum \text{Power}(X_i) \geq \text{Demand}$

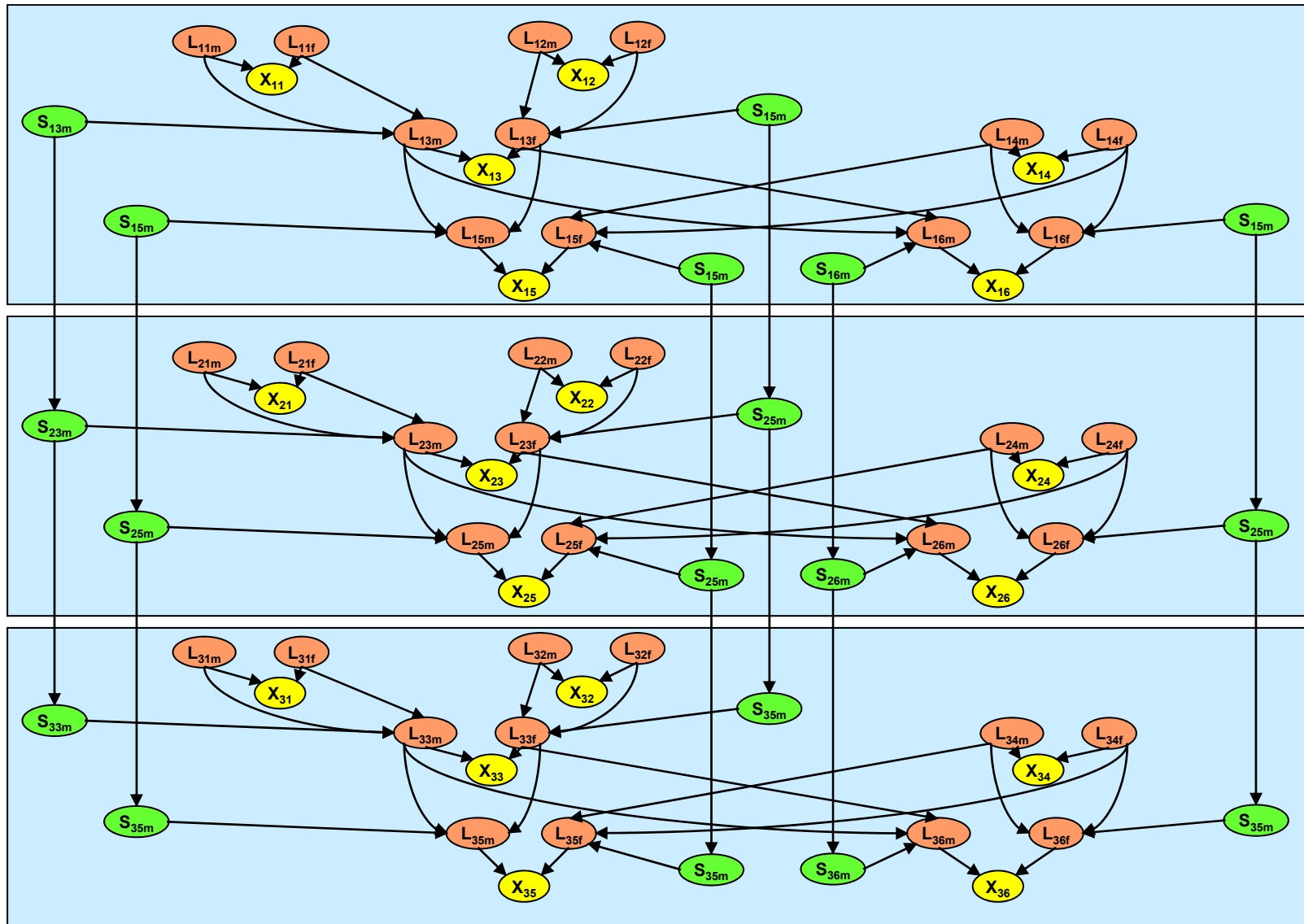
Objective : minimize $\text{TotalFuelCost}(X_1, \dots, X_N)$

Linkage Analysis



- 6 individuals
- Haplotype: {2, 3}
- Genotype: {6}
- Unknown

Pedigree: 6 people, 3 markers

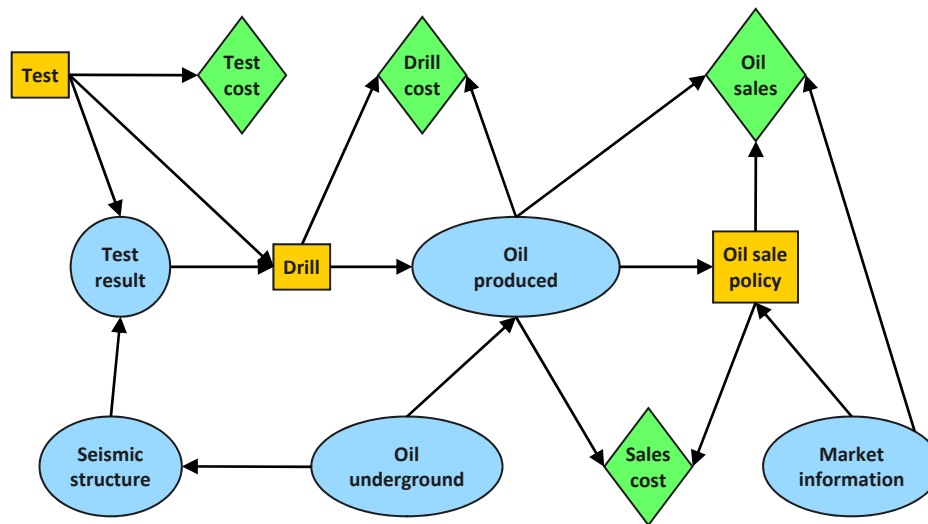


Influence Diagrams

Influence diagram $ID = (X, D, P, R)$.

Task: find optimal policy:

$$E = \max_{\Delta=(\delta_1, \dots, \delta_m)} \sum_{x=(x_1, \dots, x_n)} \prod_i P_i(x) u(x)$$



Chance variables: $X = X_1, \dots, X_n$ over domains.

Decision variables: $D = D_1, \dots, D_m$

CPT's for chance variables: $P_i = P(X_i \mid pa_i), i = 1, \dots, n$

Reward components: $R = \{r_1, \dots, r_j\}$

Utility function: $u = \sum_i r_i$

Graphical Models

- A graphical model $(\mathbf{X}, \mathbf{D}, \mathbf{F})$:

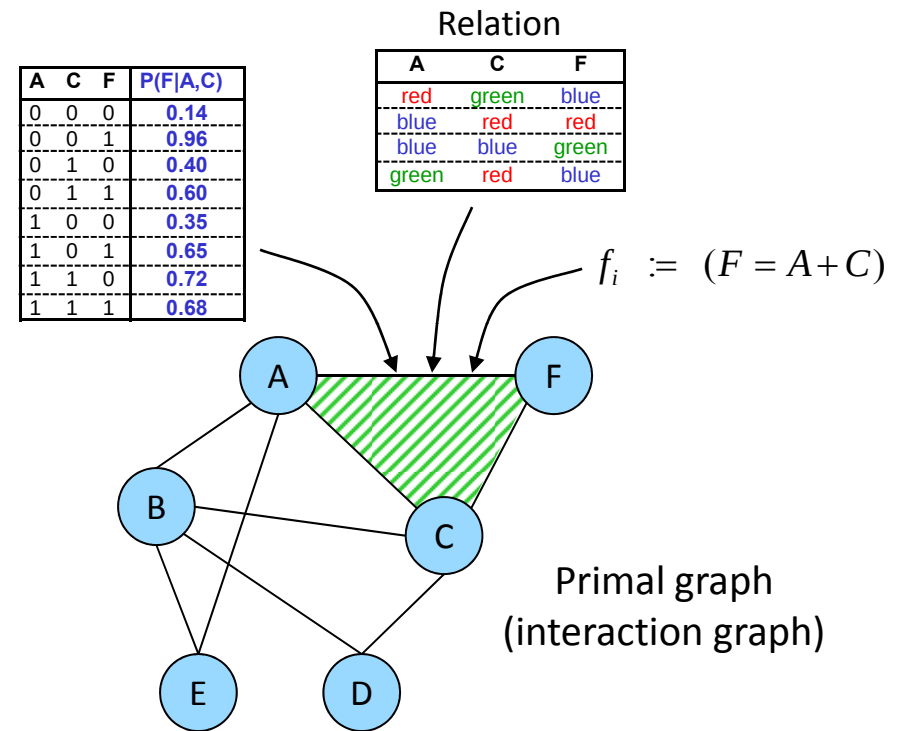
- $\mathbf{X} = \{X_1, \dots, X_n\}$ variables
- $\mathbf{D} = \{D_1, \dots, D_n\}$ domains
- $\mathbf{F} = \{f_1, \dots, f_m\}$ functions

- Operators:

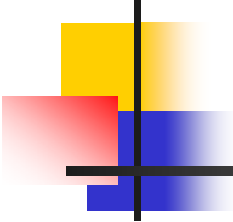
- combination
- elimination (projection)

- Tasks:

- **Belief updating:** $\sum_{x-y} \prod_i P_i$
- **MPE:** $\max_x \prod_j P_j$
- **CSP:** $\prod_{x \times_j} C_j$
- **Max-CSP:** $\min_x \sum_j f_j$



- All these tasks are NP-hard
 - exploit problem structure
 - identify special cases
 - approximate



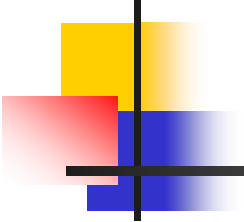
Sample Domains for Graphical Models

- Web Pages and Link Analysis
- Communication Networks (Cell phone Fraud Detection)
- Natural Language Processing (e.g. Information Extraction and Semantic Parsing)
- Battle-space Awareness
- Epidemiological Studies
- Citation Networks
- Intelligence Analysis (Terrorist Networks)
- Financial Transactions (Money Laundering)
- Computational Biology
- Object Recognition and Scene Analysis

...

Type of constrained optimization:

- Weighted CSPs, Max-CSPs, Max-SAT
- Most Probable Explanation (MPE)
- Linear Integer Programs



Outline

- Introduction

- Graphical models

- Solution Techniques

- Inference

- Search

- Compilation: AND/OR Decision Diagrams

- Software

Solution Techniques

AND/OR search

*Time: $\exp(\text{treewidth} * \log n)$*

Space: linear

Space: $\exp(\text{treewidth})$

Time: $\exp(\text{treewidth})$

Time: $\exp(\text{treewidth})$

Space: $\exp(\text{treewidth})$

Inference (Elimination)

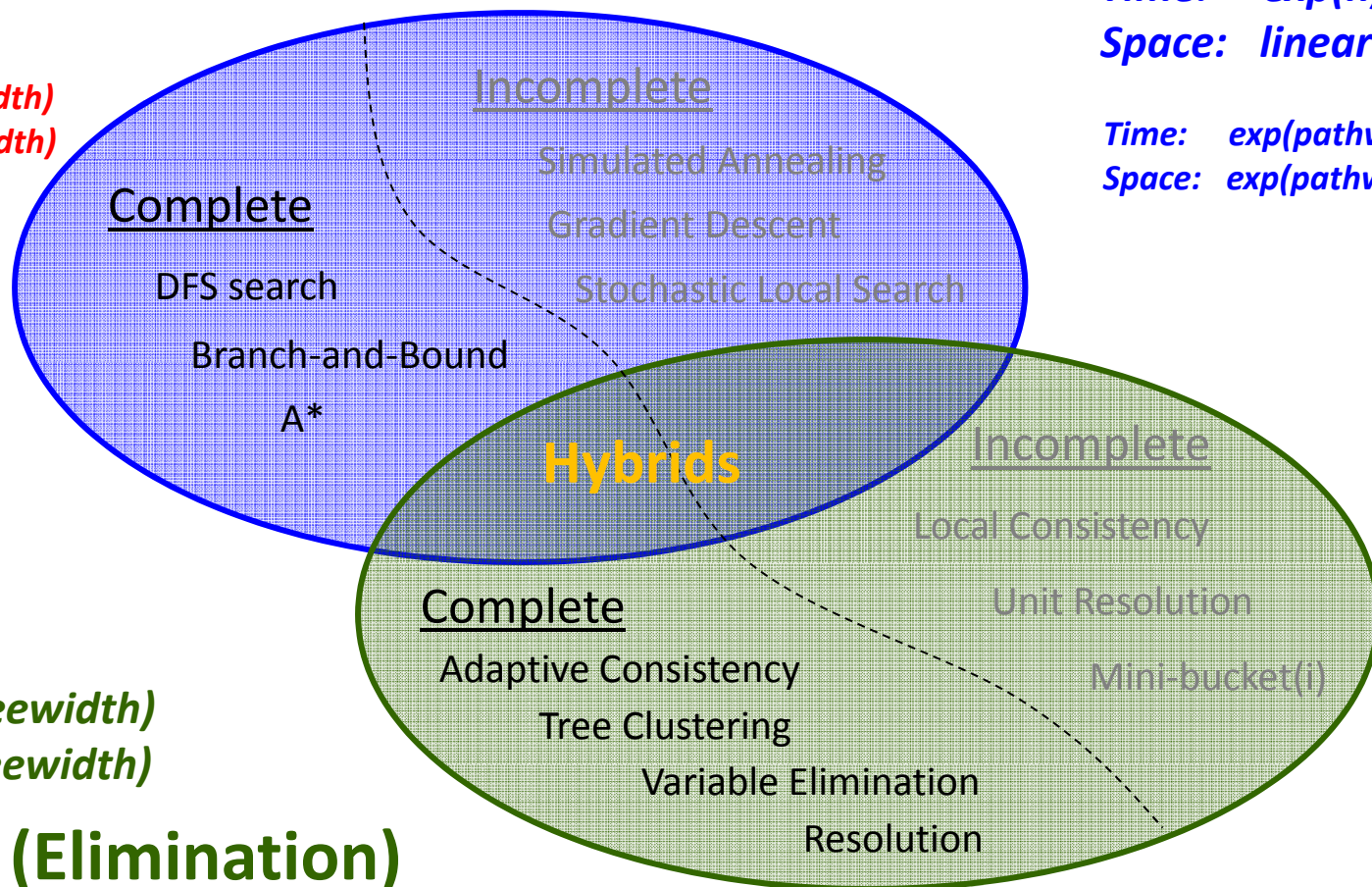
Search (Conditioning)

Time: $\exp(n)$

Space: linear

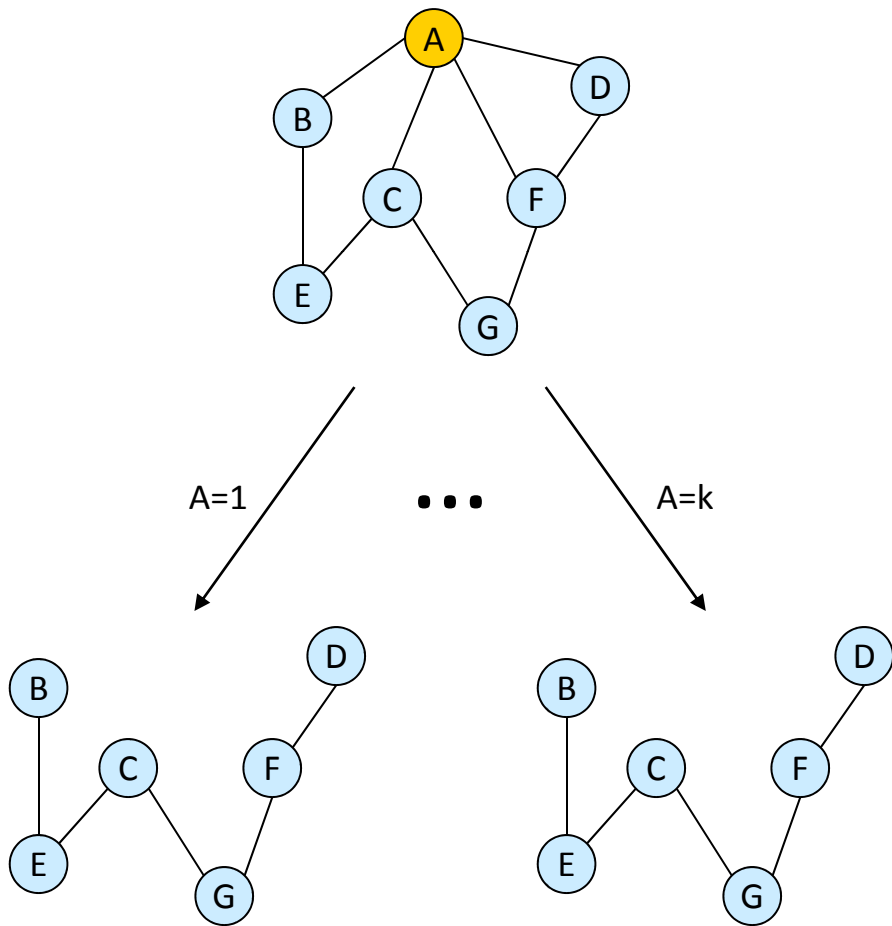
Time: $\exp(\text{pathwidth})$

Space: $\exp(\text{pathwidth})$



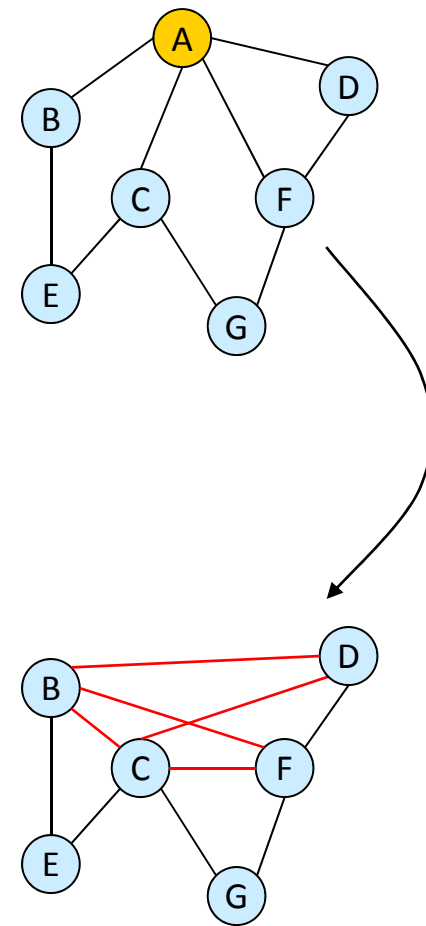
Search vs. Inference

Search (conditioning)

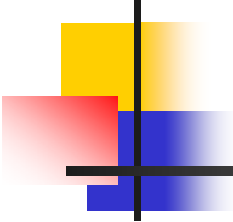


k "sparser" problems

Inference (elimination)



1 "denser" problem



Combination of Cost Functions

A	B	f(A,B)
b	b	6
b	g	0
g	b	0
g	g	6

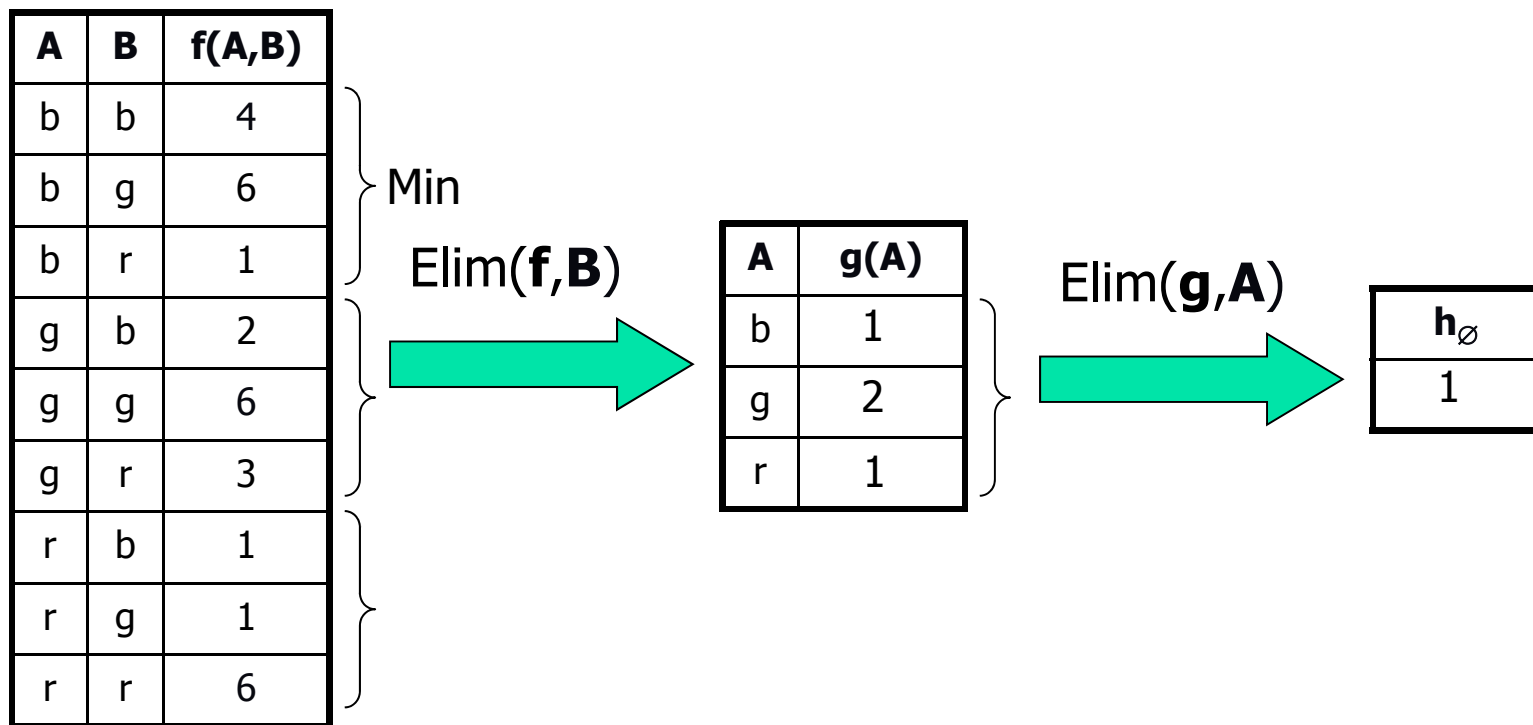
+

B	C	f(B,C)
b	b	6
b	g	0
g	b	0
g	g	6

A	B	C	f(A,B,C)
b	b	b	12
b	b	g	6
b	g	b	0
b	g	g	6
g	b	b	6
g	b	g	0
g	g	b	6
g	g	g	12

= 0 + 6

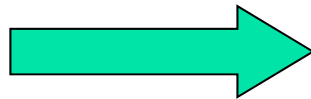
Elimination in a Cost Function



Conditioning a Cost Function

A	B	f(A,B)
b	b	6
b	g	0
b	r	3
g	b	0
g	g	6
g	r	0
r	b	0
r	g	0
r	r	6

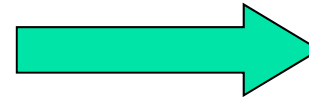
Assign($\mathbf{f}_{AB}, A, b$)



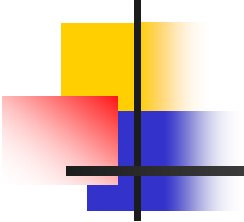
$g(B)$

3

Assign($\mathbf{g}, B, r$)



$h_{\emptyset}$



Outline

- Introduction

- Inference

- Exact: Variable elimination, bucket elimination
- Approximate: Belief propagation

- Search

- Compilation: AND/OR Decision Diagrams

- Software

Outline

■ Introduction

■ Inference

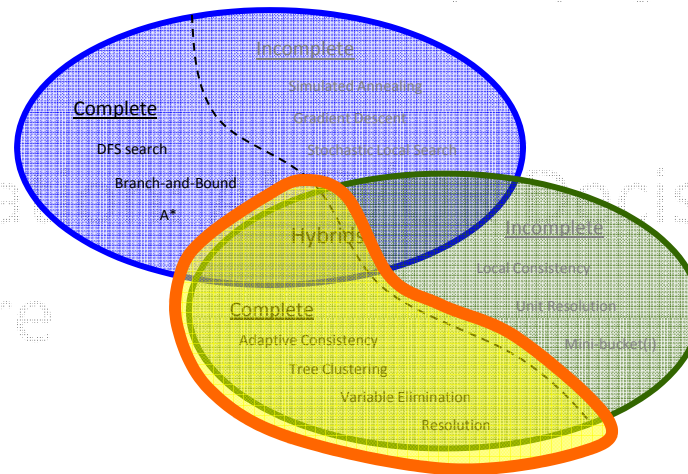
■ Exact: Variable elimination, bucket elimination

■ Approximate: Belief propagation

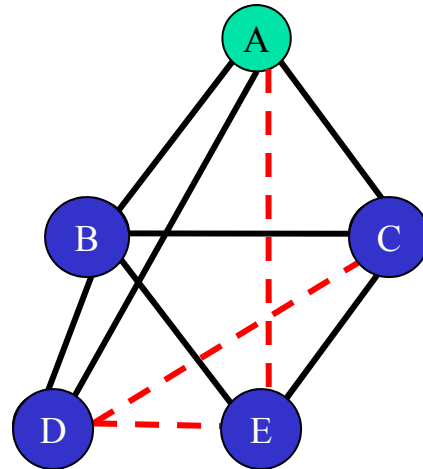
■ Search

■ Compilation

■ Software



Computing the Optimal Cost Solution



Constraint graph

$$\mathbf{OPT} = \min_{e=0, d, c, b} \underbrace{f(a, \mathbf{b}) + f(a, c) + f(a, d)}_{\text{solid}} + \underbrace{f(\mathbf{b}, c) + f(\mathbf{b}, d) + f(\mathbf{b}, e) + f(c, e)}_{\text{dashed}}$$

Combination

$$\min_{e=0} \min_d f(a, d) + \min_c f(a, c) + f(c, e) + \underbrace{\min_b f(a, b) + f(b, c) + f(b, d) + f(b, e)}_{h^B(a, d, c, e)}$$

Variable Elimination

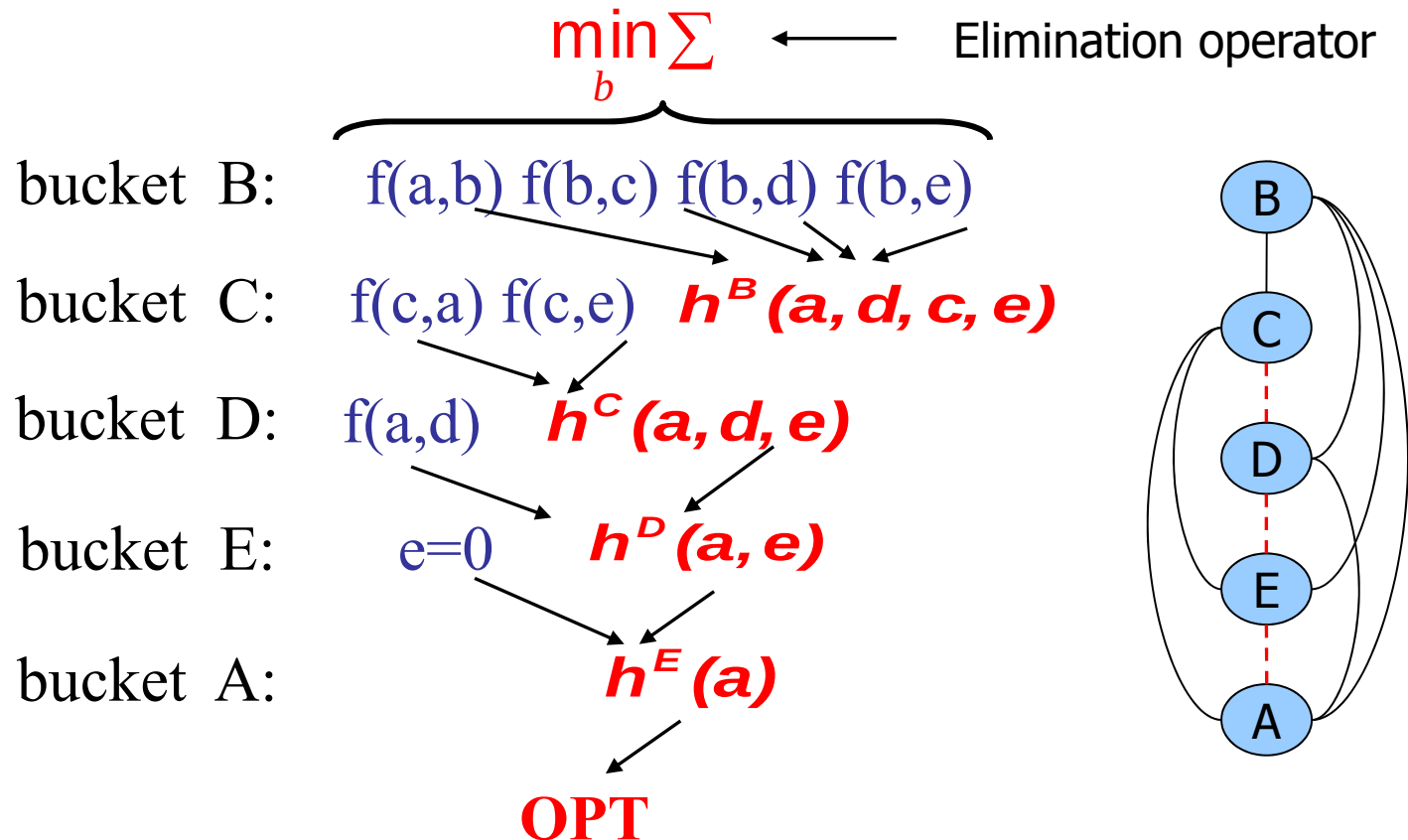
Finding

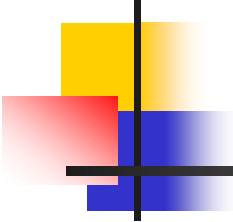
$$OPT = \min_{X_1, \dots, X_n} \sum_{j=1}^r f_j(X)$$

Algorithm **elim-opt** (Dechter, 1996)

Non-serial Dynamic Programming (Bertele & Briochi, 1973)

$$OPT = \min_{a,e,d,c,b} F(a,b) + F(a,c) + F(a,d) + F(b,c) + F(b,d) + F(b,e) + F(c,e)$$





Generating the Optimal Assignment

5. $b' = \arg \min_b f(a', b) + f(b, c') +$
 $+ f(b, d') + f(b, e')$

4. $c' = \arg \min_c f(c, a') + f(c, e') +$
 $+ h^B(a', d', c, e')$

3. $d' = \arg \min_d f(a', d) + h^C(a', d, e')$

2. $e' = 0$

1. $a' = \arg \min_a h^E(a)$

B: $f(a, b) f(b, c) f(b, d) f(b, e)$

C: $f(c, a) f(c, e) \quad h^B(a, d, c, e)$

D: $f(a, d) \quad h^C(a, d, e)$

E: $e=0 \quad h^D(a, e)$

A: $h^E(a)$

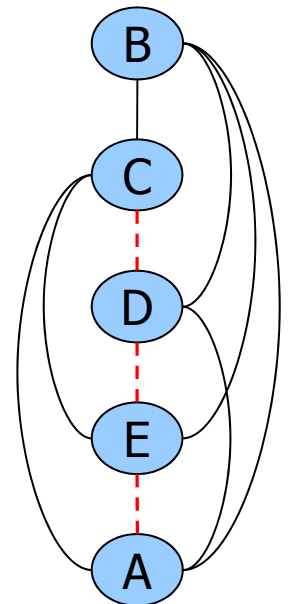
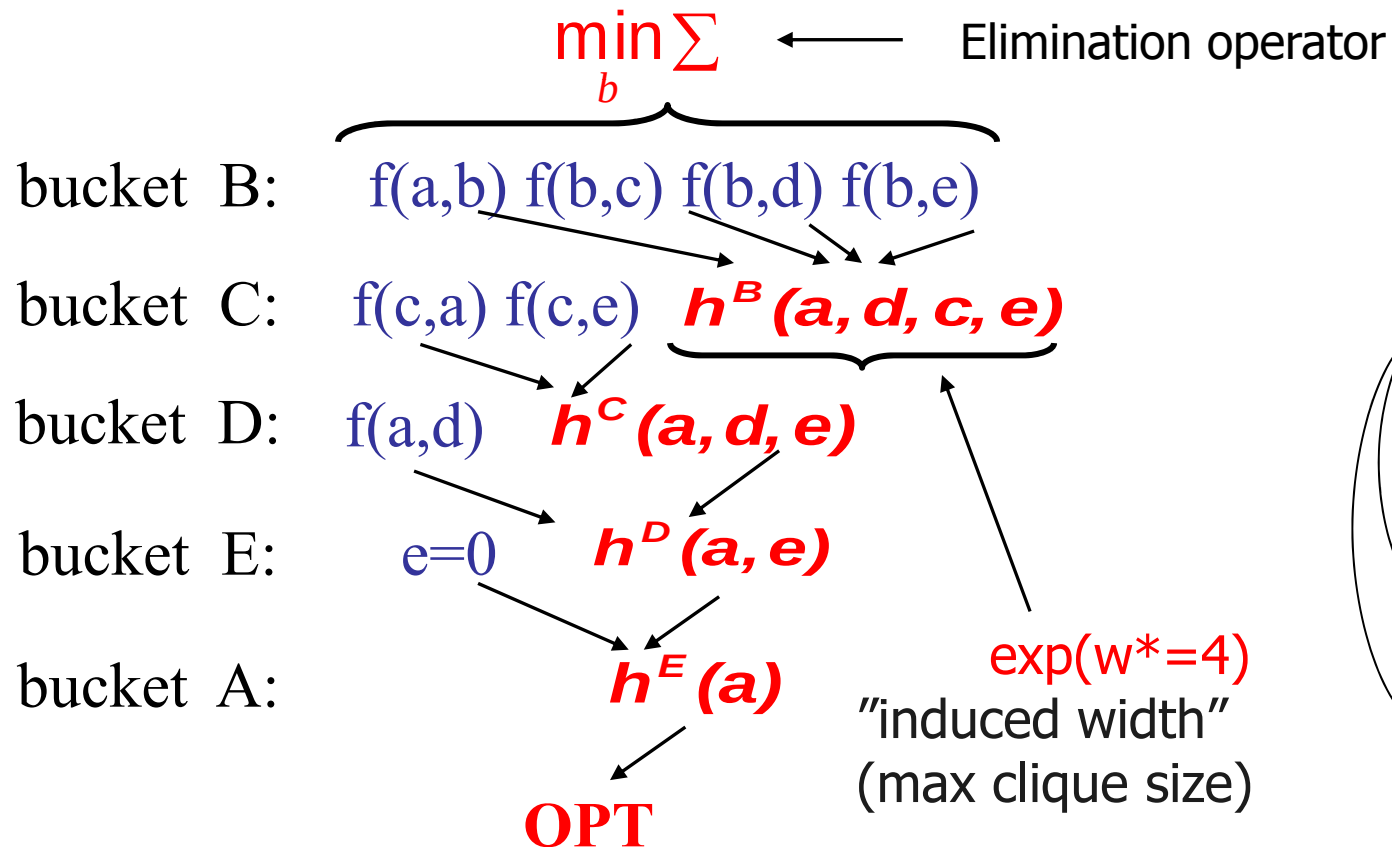
Return (a', b', c', d', e')

Complexity

Algorithm **elim-opt** (Dechter, 1996)

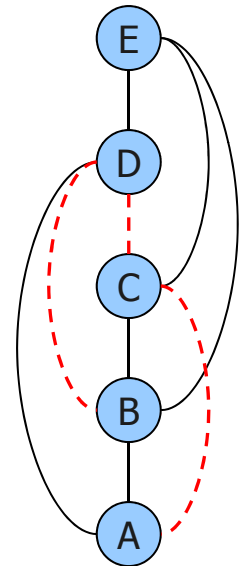
Non-serial Dynamic Programming (Bertele & Briochi, 1973)

$$OPT = \min_{a,e,d,c,b} F(a,b) + F(a,c) + F(a,d) + F(b,c) + F(b,d) + F(b,e) + F(c,e)$$



Induced-width

- **Width** along ordering d , $w(d)$:
 - max # of previous neighbors (parents)
- **Induced width** along ordering d , $w^*(d)$:
 - The width in the ordered **induced graph**, obtained by connecting “parents” of each node X , recursively from top to bottom



Complexity of Bucket Elimination

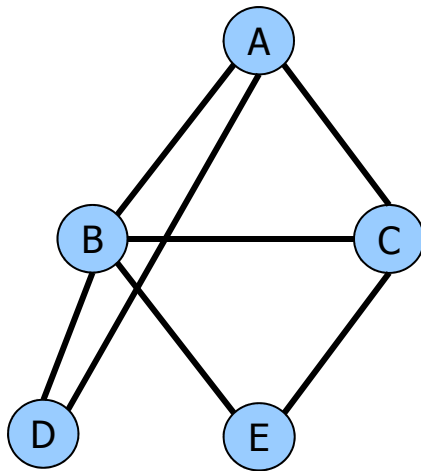
Bucket-Elimination is **time** and **space**

$$O(r \exp(w^*(d)))$$

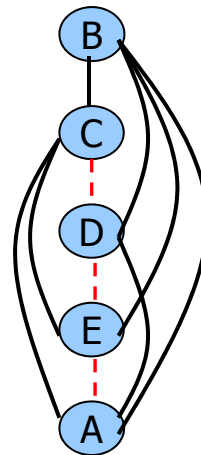
$w^*(d)$ – the induced width of the primal graph along ordering d

r = number of functions

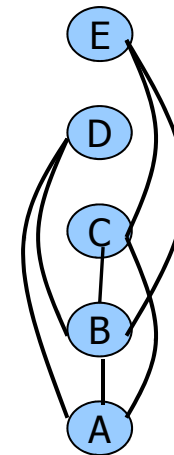
The effect of the ordering:



constraint graph



$$w^*(d_1) = 4$$



$$w^*(d_2) = 2$$

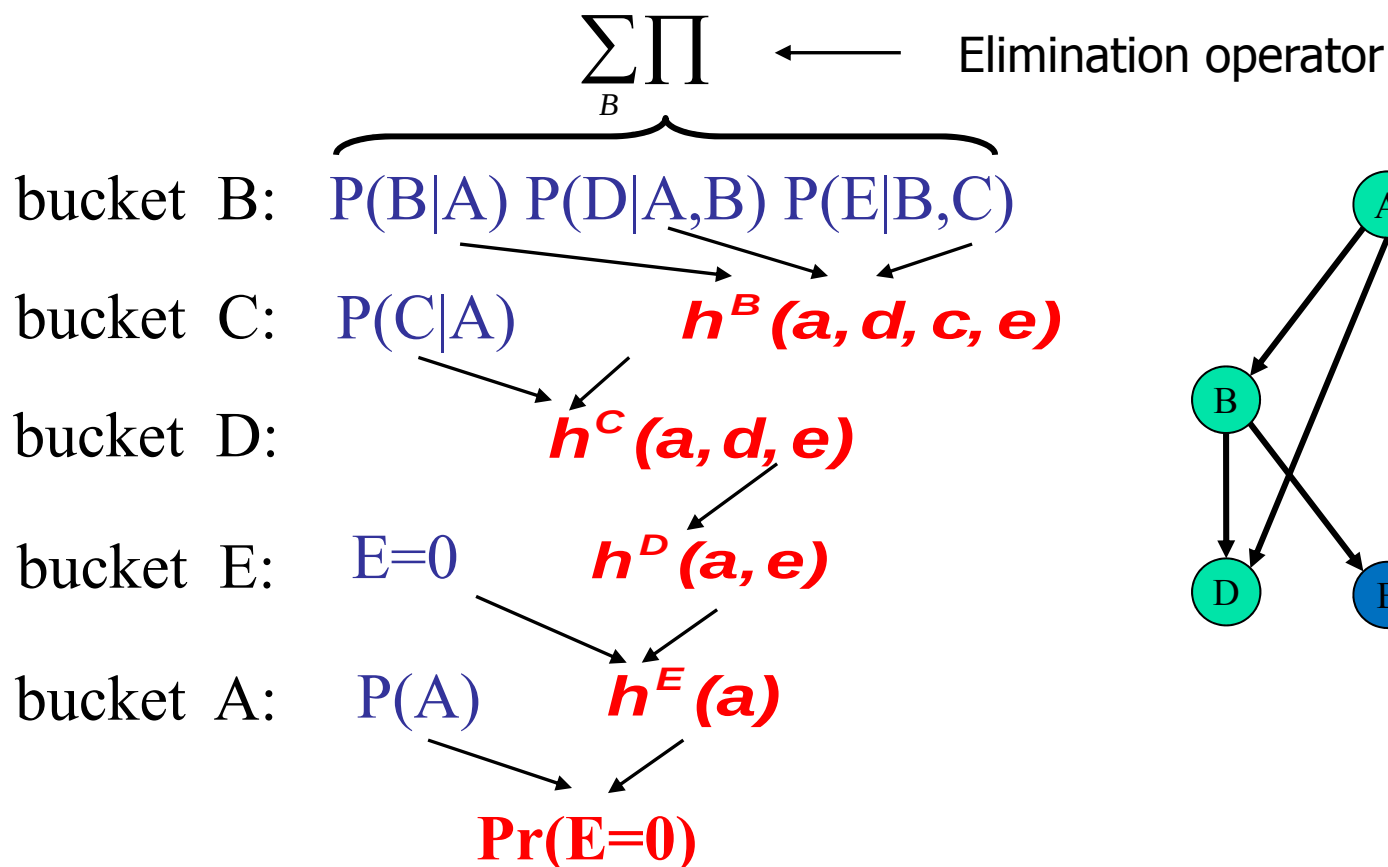
Finding smallest induced-width is hard!

Finding

$$P(\text{evidence}) = \sum_{X_1, \dots, X_n} \prod_{j=1}^n P(X_j \mid pa_j)$$

Algorithm **elim-bel** (Dechter, 1996)

$$P(E = 0) = \sum_{A, E=0, D, C, B} P(A) \cdot P(B \mid A) \cdot P(C \mid A) \cdot P(D \mid A, B) \cdot P(E \mid B, C)$$



Outline

■ Introduction

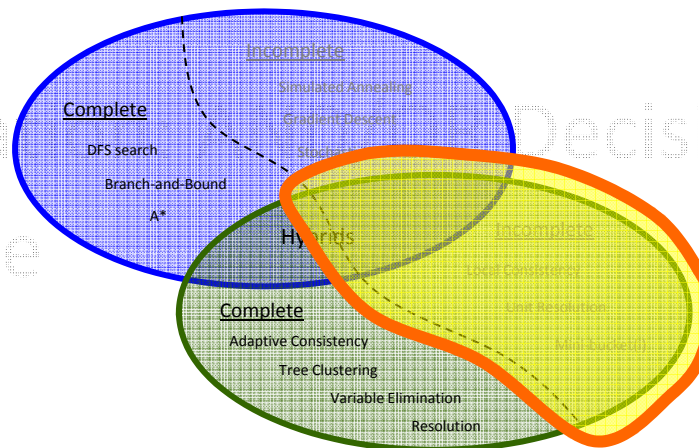
■ Inference

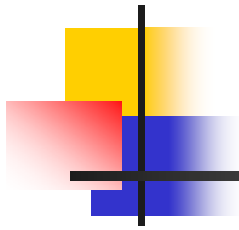
- Exact: Variable elimination, bucket elimination
- Approximate: Belief propagation

■ Search

■ Compilation

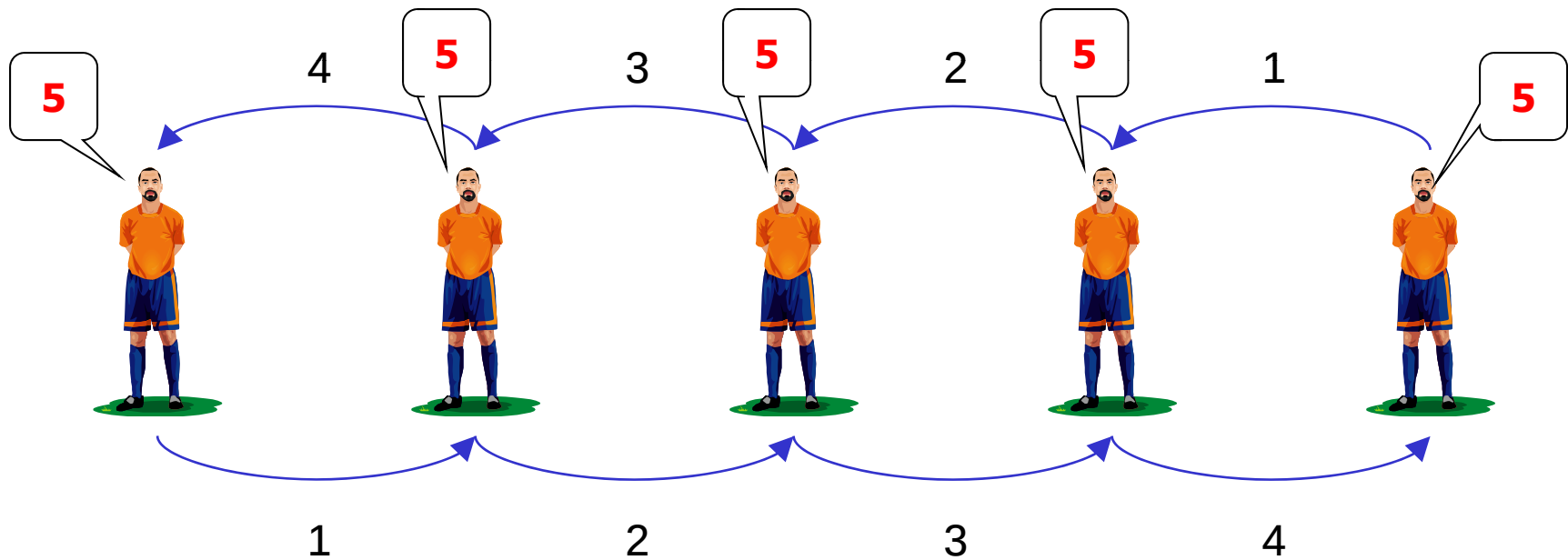
■ Software





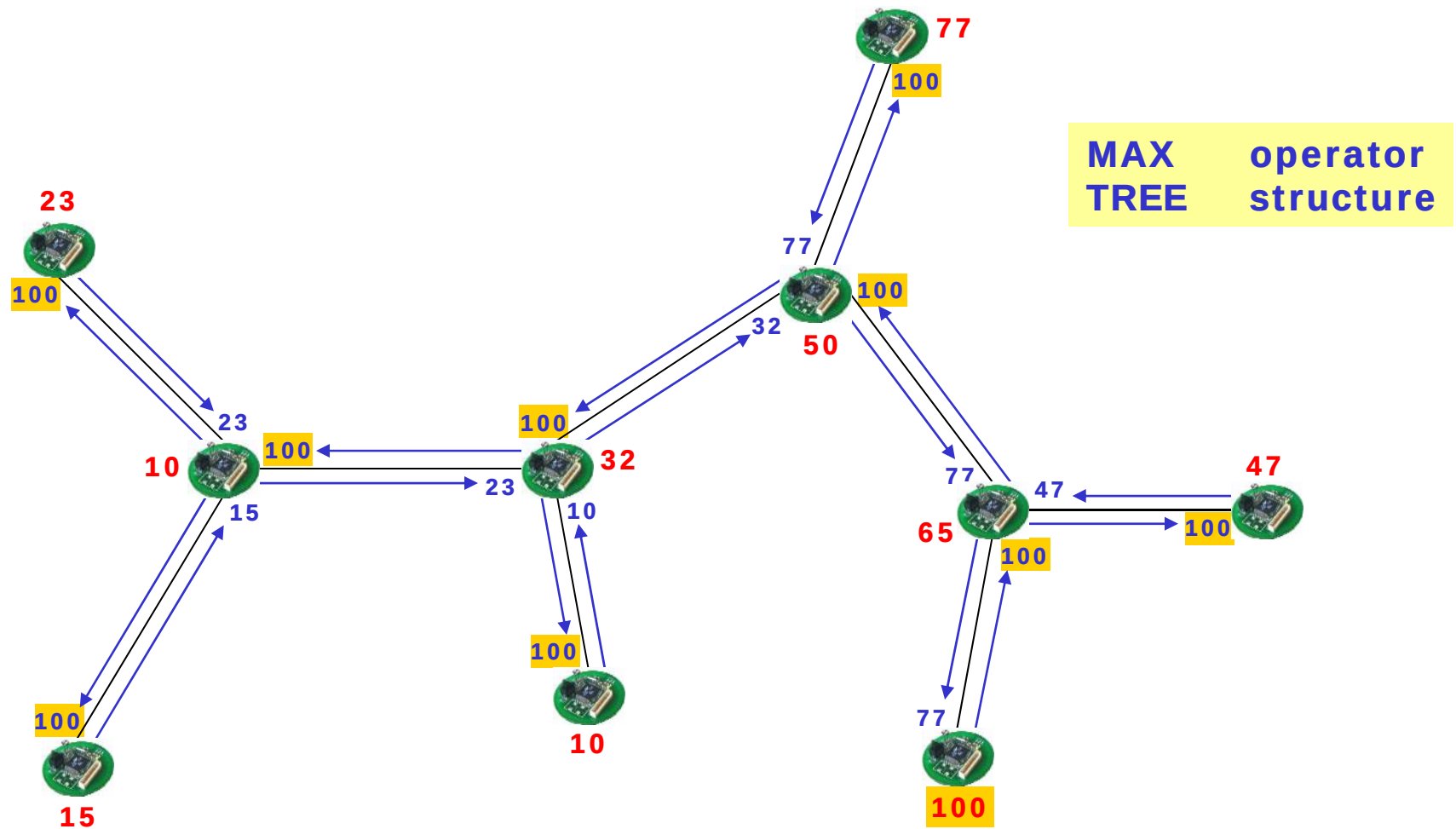
Counting

SUM operator
CHAIN structure



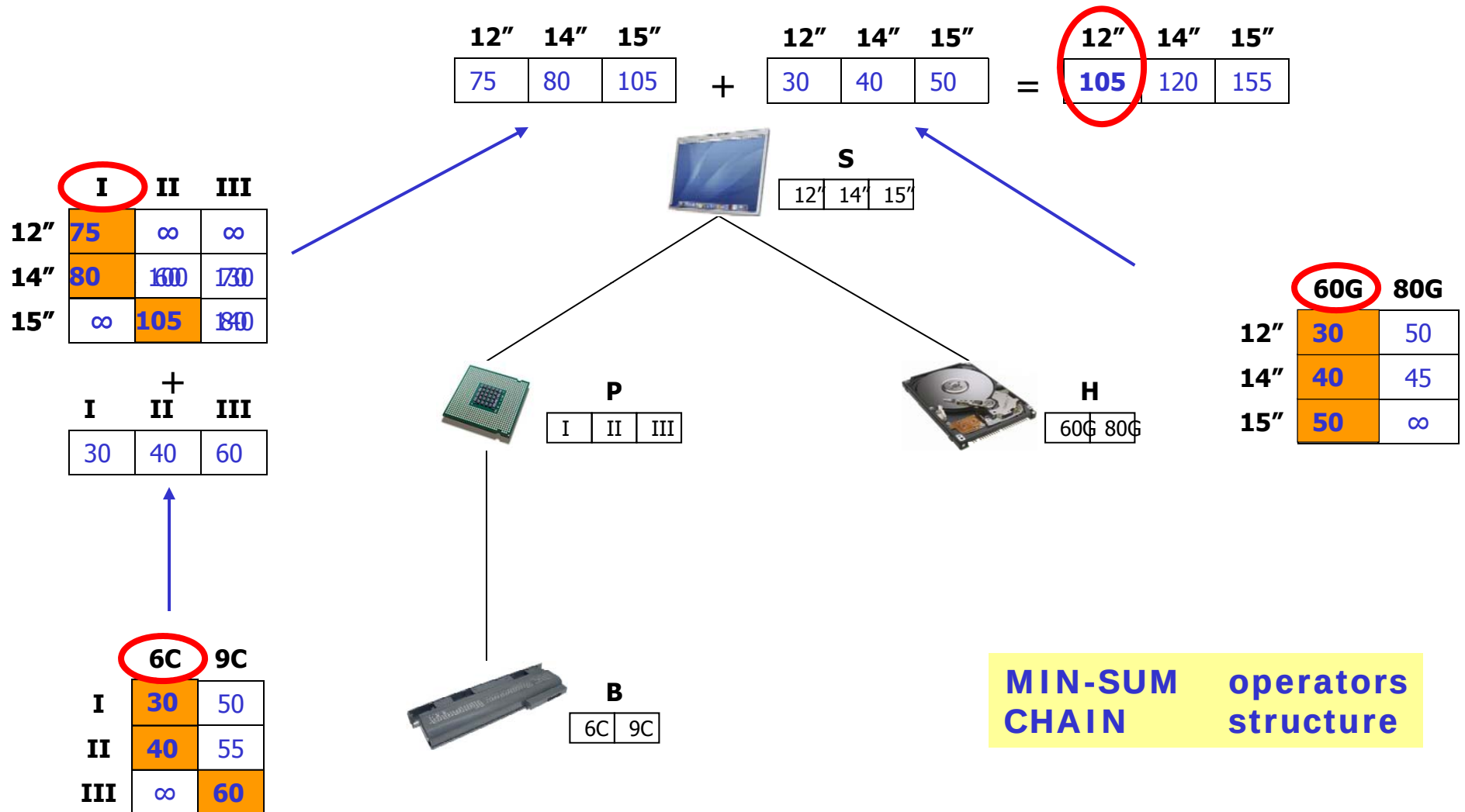
How many people?

Maximization



What is the maximum?

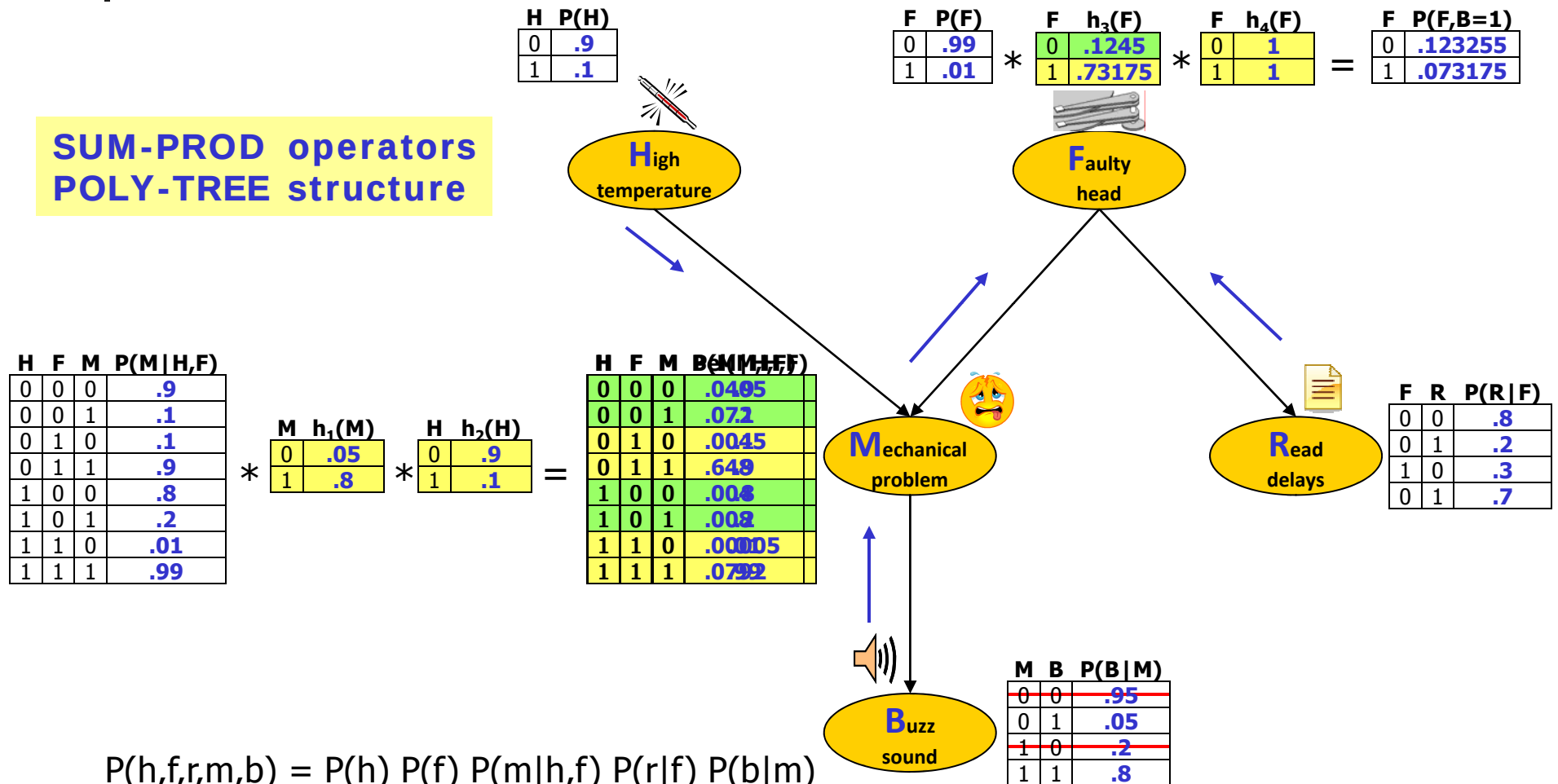
Min-Cost Assignment



What is minimum cost configuration?

Belief Updating

SUM-PROD operators
POLY-TREE structure



$$P(h, f, r, m, b) = P(h) P(f) P(m|h, f) P(r|f) P(b|m)$$

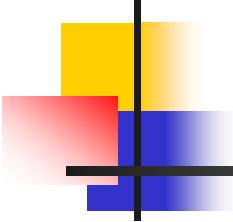
$$P(F | B=1) = ?$$

$$P(B=1) = .19643$$

$$P(F=1|B=1) = .3725$$

Probability of evidence

Updated belief



Belief Propagation

[Pearl, 1988]

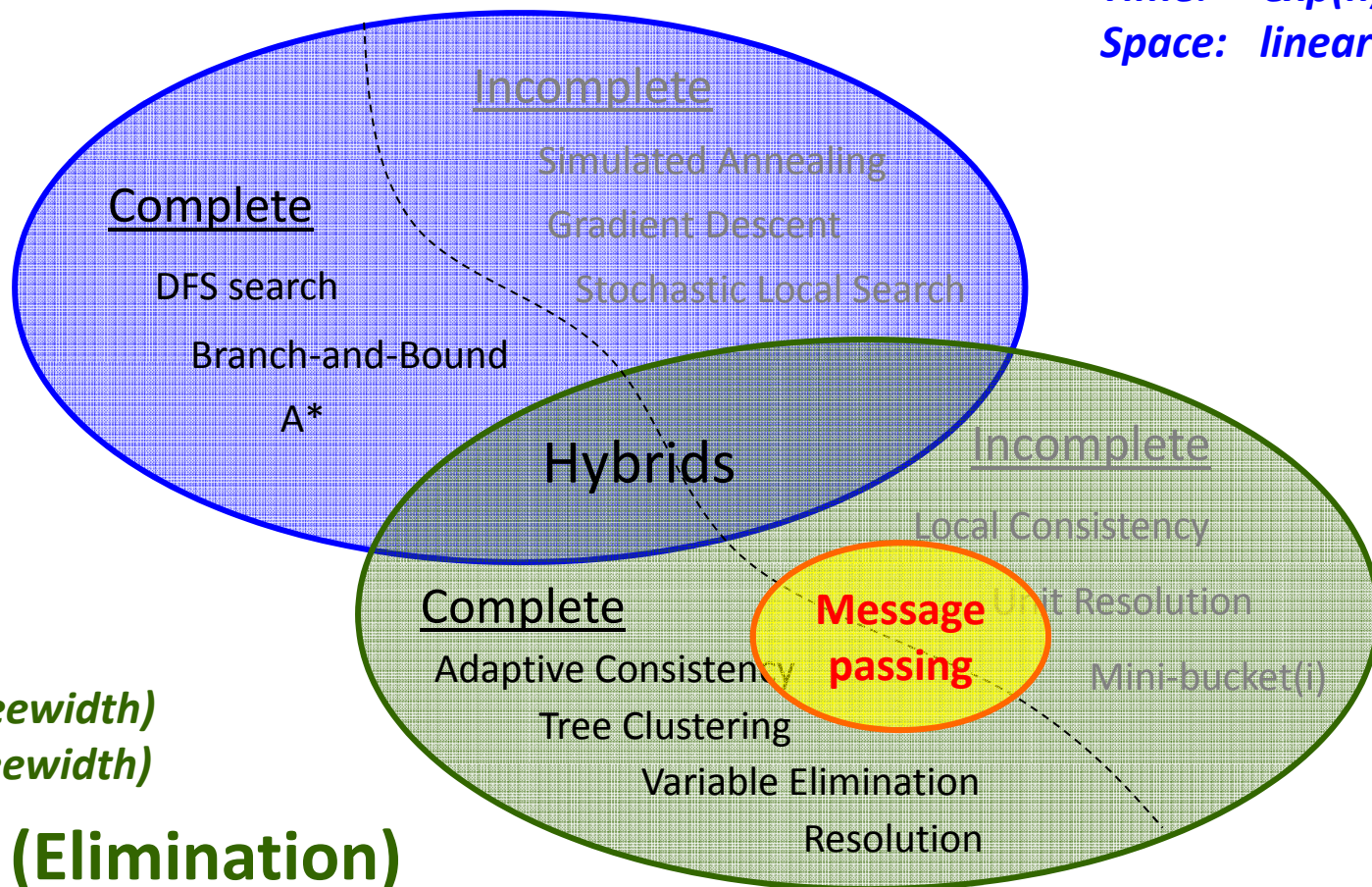
- Instances of **tree message passing** algorithm
- **Exact** for trees
- **Linear** in the input size
- Importance:
 - One of the first algorithms for inference in Bayesian networks
 - Gives a cognitive dimension to its computations
 - Basis for conditioning algorithms for arbitrary Bayesian network
 - Basis for **Loopy Belief Propagation** (approximate algorithms)

Solution Techniques

Search (Conditioning)

Time: $\exp(n)$

Space: linear

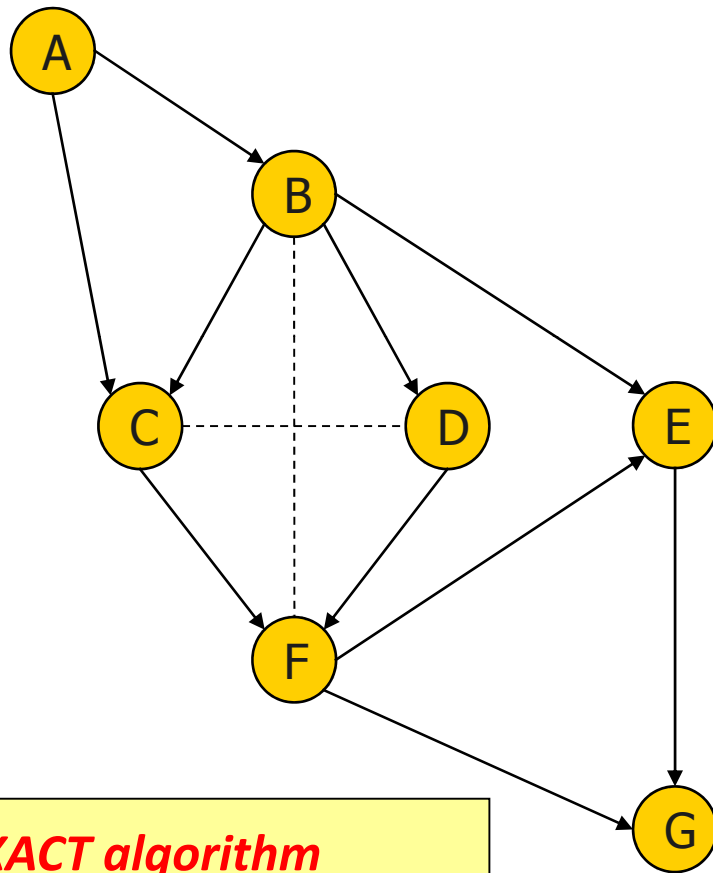


Time: $\exp(\text{treewidth})$

Space: $\exp(\text{treewidth})$

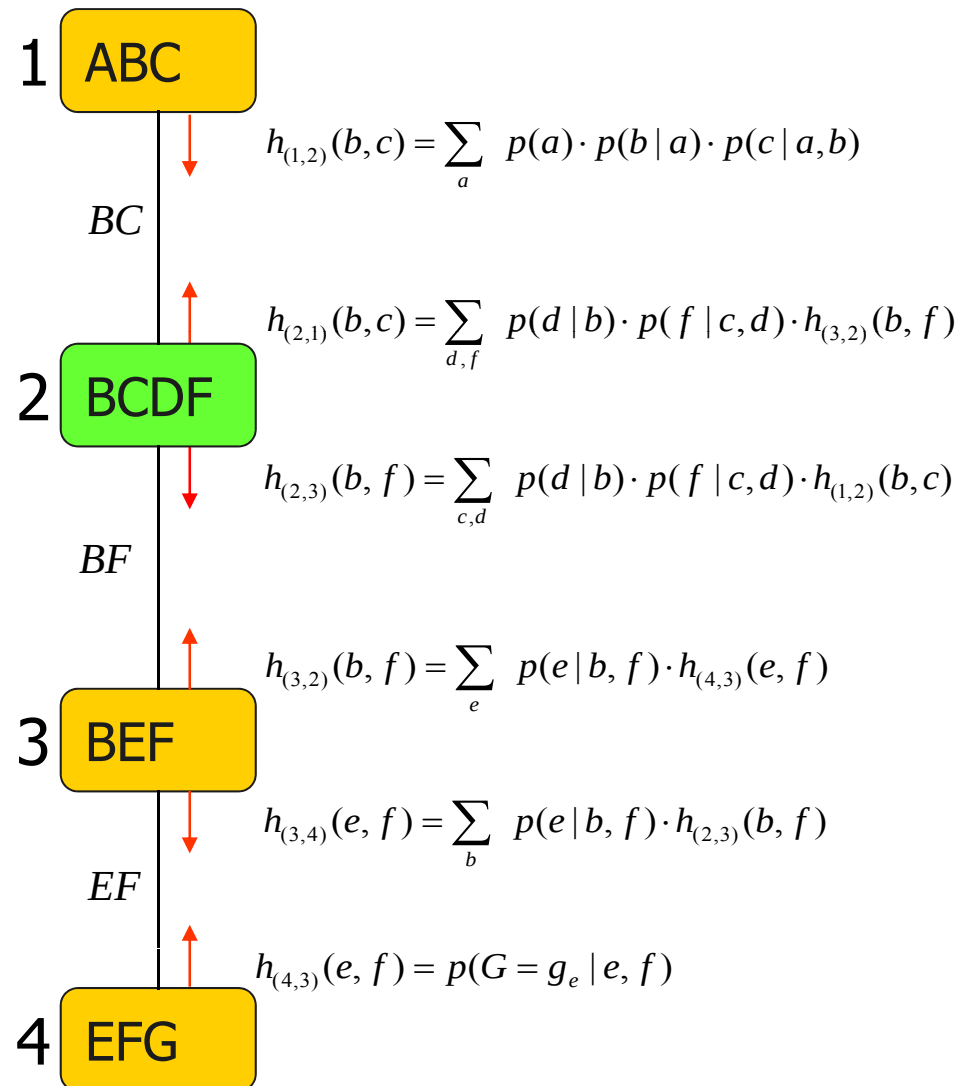
Inference (Elimination)

Join-Tree Clustering



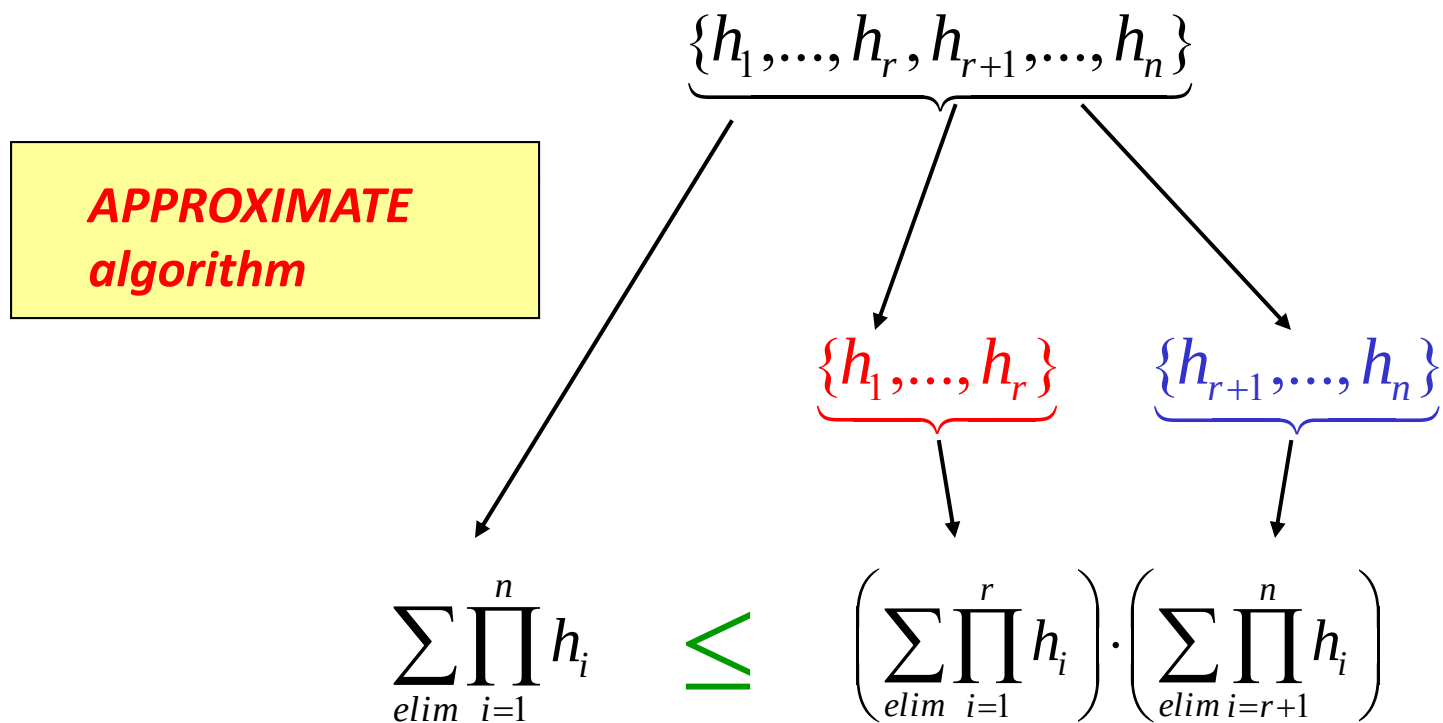
EXACT algorithm

Time and space:
 $\exp(\text{cluster size}) =$
 $\exp(\text{treewidth})$



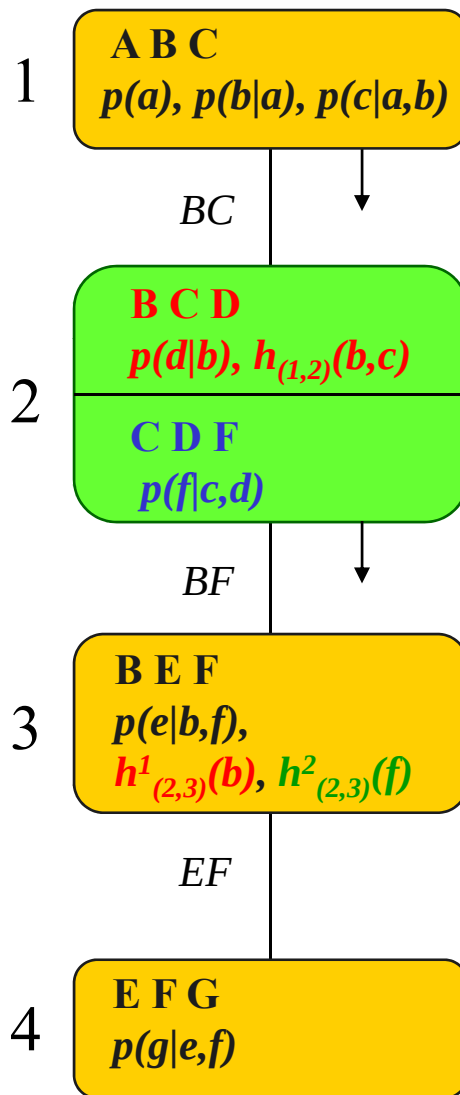
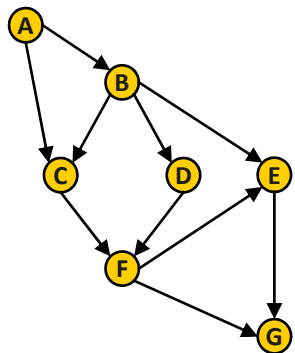
Mini-Clustering

Split a cluster into mini-clusters $\Rightarrow$ bound complexity



Exponential complexity decrease $O(e^n) \rightarrow O(e^{\text{var}(r)}) + O(e^{\text{var}(n-r)})$

Mini-Clustering, i-bound=3



$$h^1_{(1,2)}(b, c) = \sum_a p(a) \cdot p(b|a) \cdot p(c|a,b)$$

$$h^1_{(2,3)}(b) = \sum_{c,d} p(d|b) \cdot h^1_{(1,2)}(b, c)$$

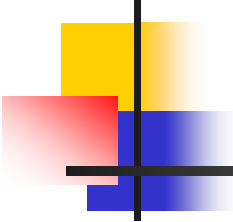
$$h^2_{(2,3)}(f) = \max_{c,d} p(f|c,d)$$

APPROXIMATE algorithm

Time and space:

$\exp(i\text{-bound})$

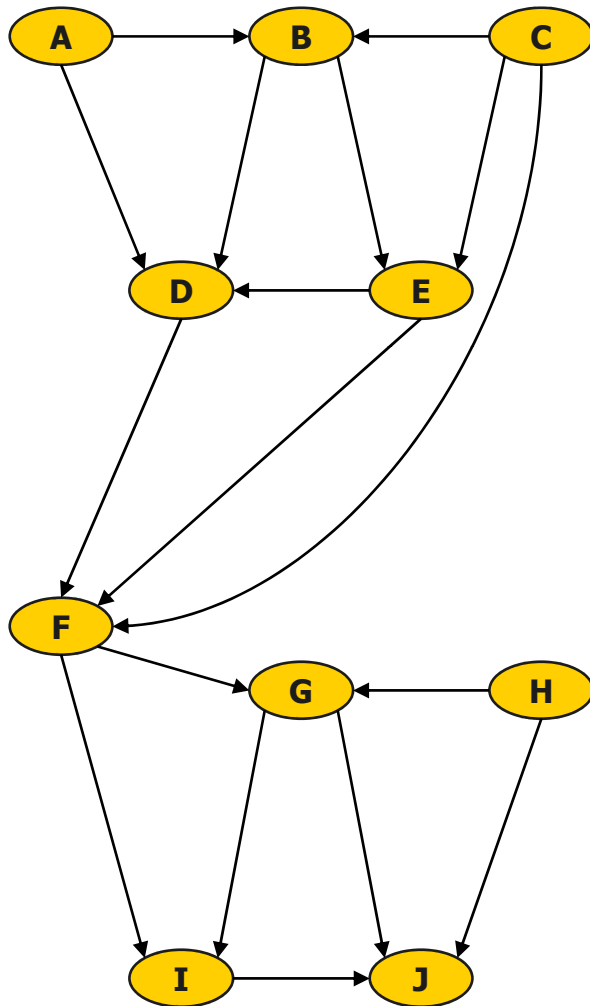
Number of variables in a mini-cluster



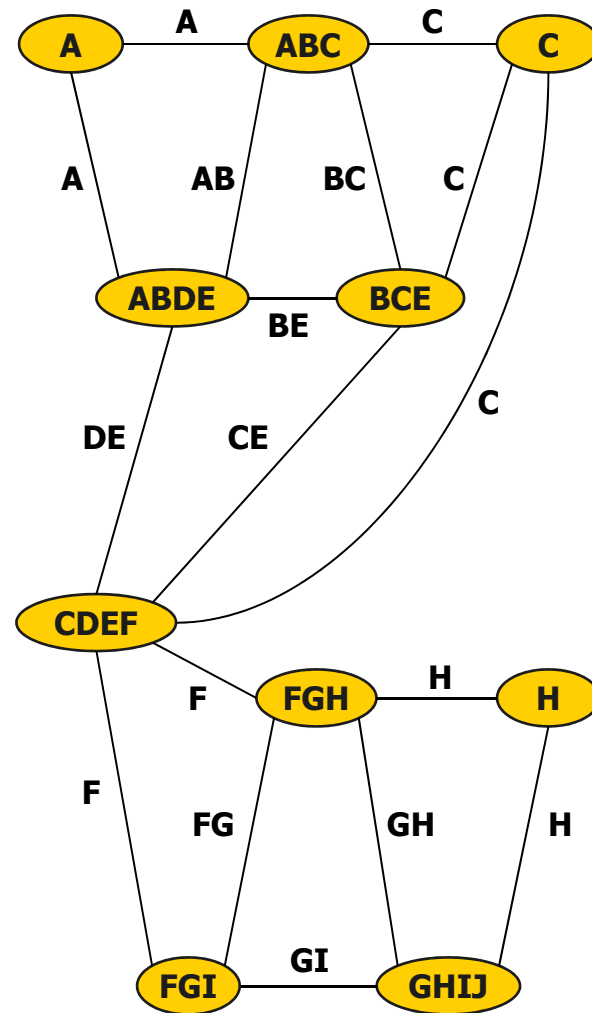
Iterative Join Graph Propagation

- Loopy Belief Propagation
 - Cyclic graphs
 - **Iterative**
 - Converges fast in practice (no guarantees though)
 - Very good approximations (e.g., turbo decoding, LDPC codes, SAT – survey propagation)
- Mini-Clustering(i)
 - Tree decompositions
 - Only two sets of messages (inward, outward)
 - **Anytime** behavior – can improve with more time by increasing the i-bound
- We want to combine:
 - Iterative virtues of Loopy BP
 - Anytime behavior of Mini-Clustering(i)

IJGP - Example



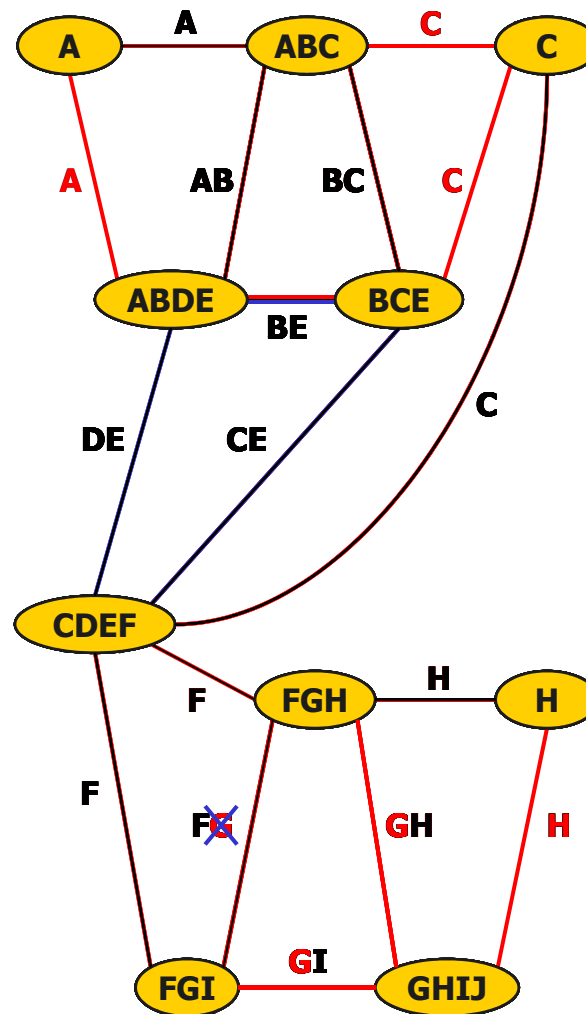
Belief network



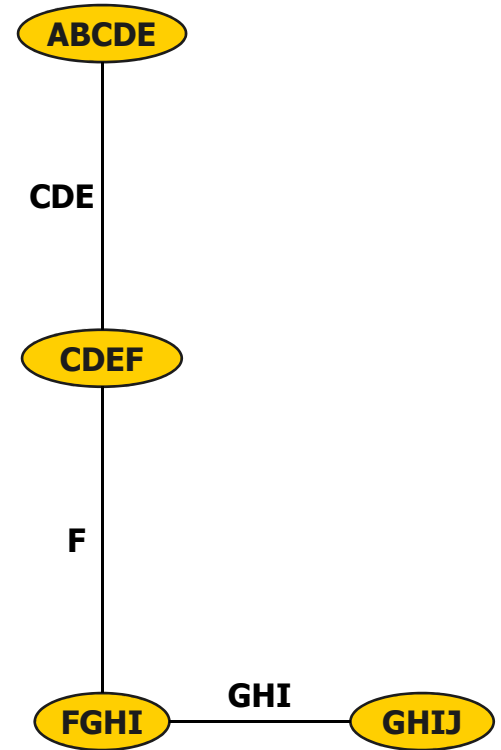
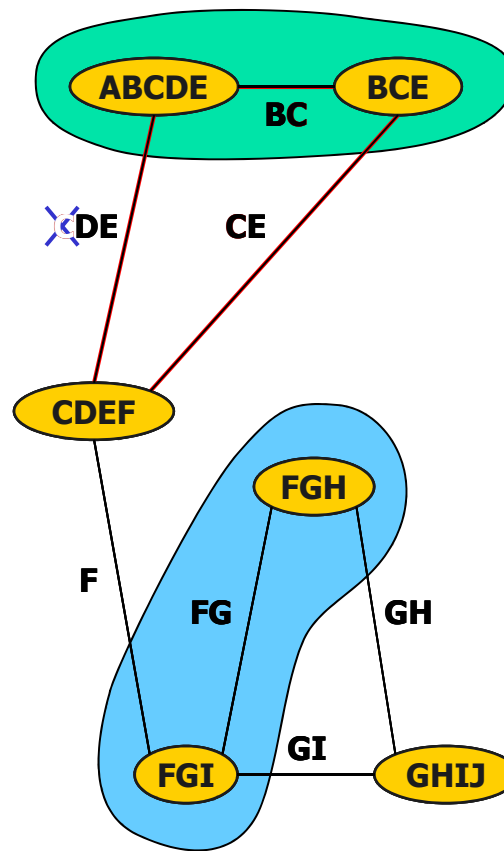
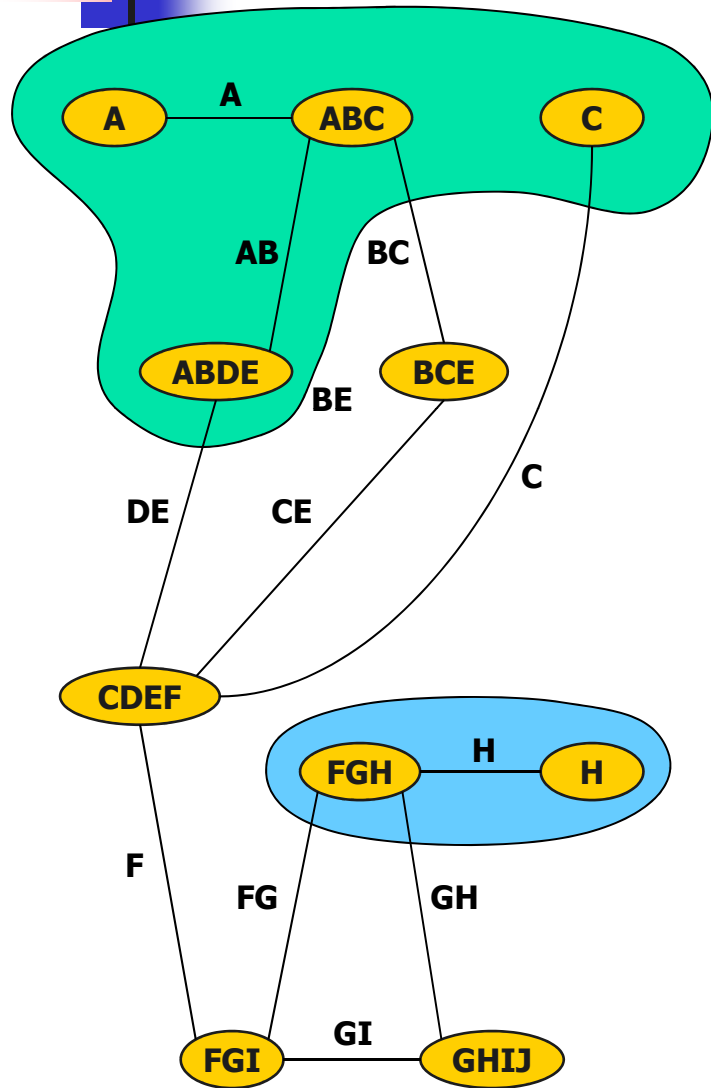
Loopy BP graph

Arc-Minimal Join-Graph

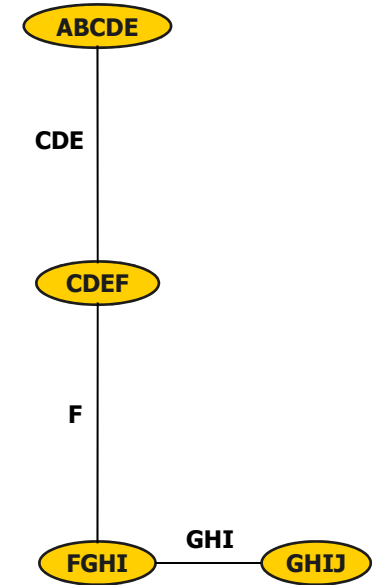
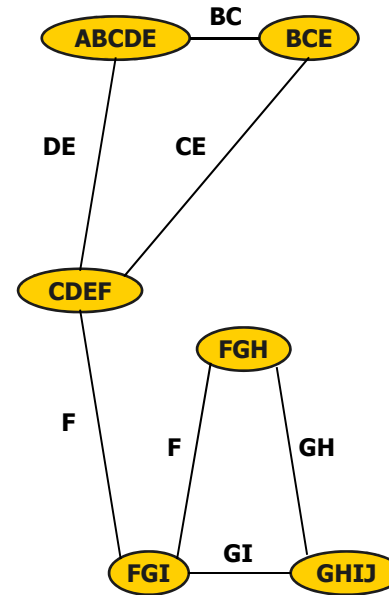
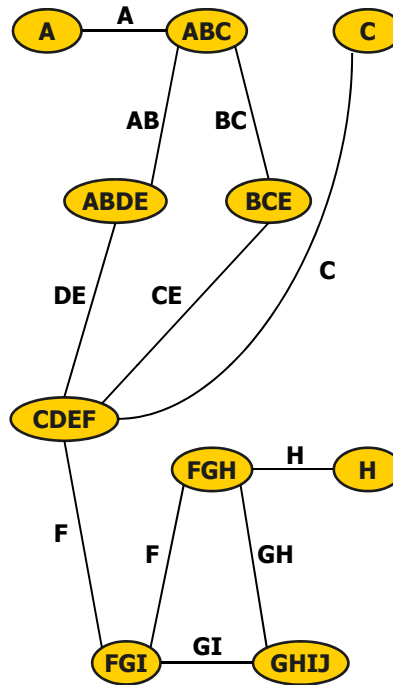
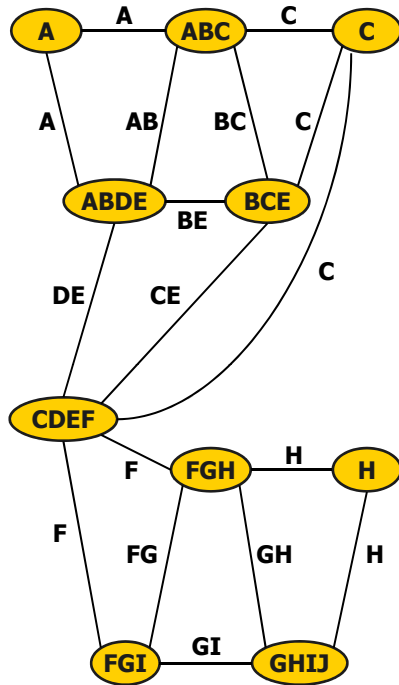
Arcs labeled with
any single variable
should form a **TREE**



Collapsing Clusters



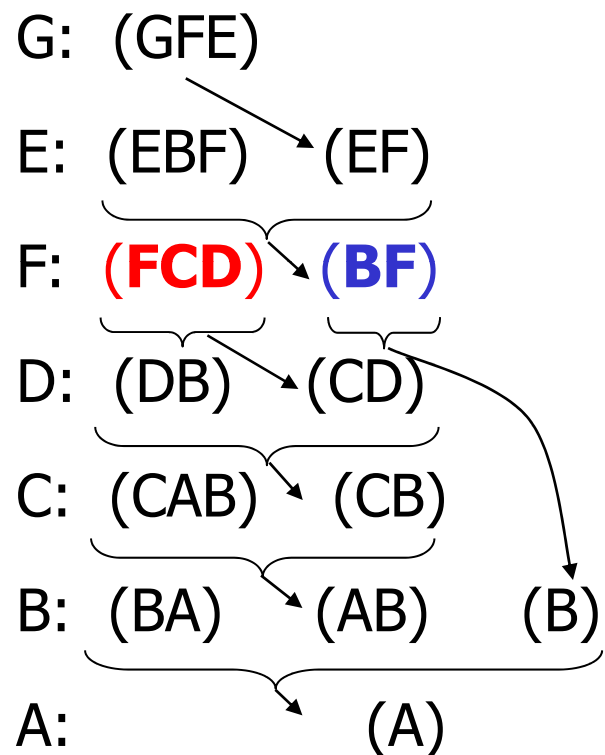
Join-Graphs



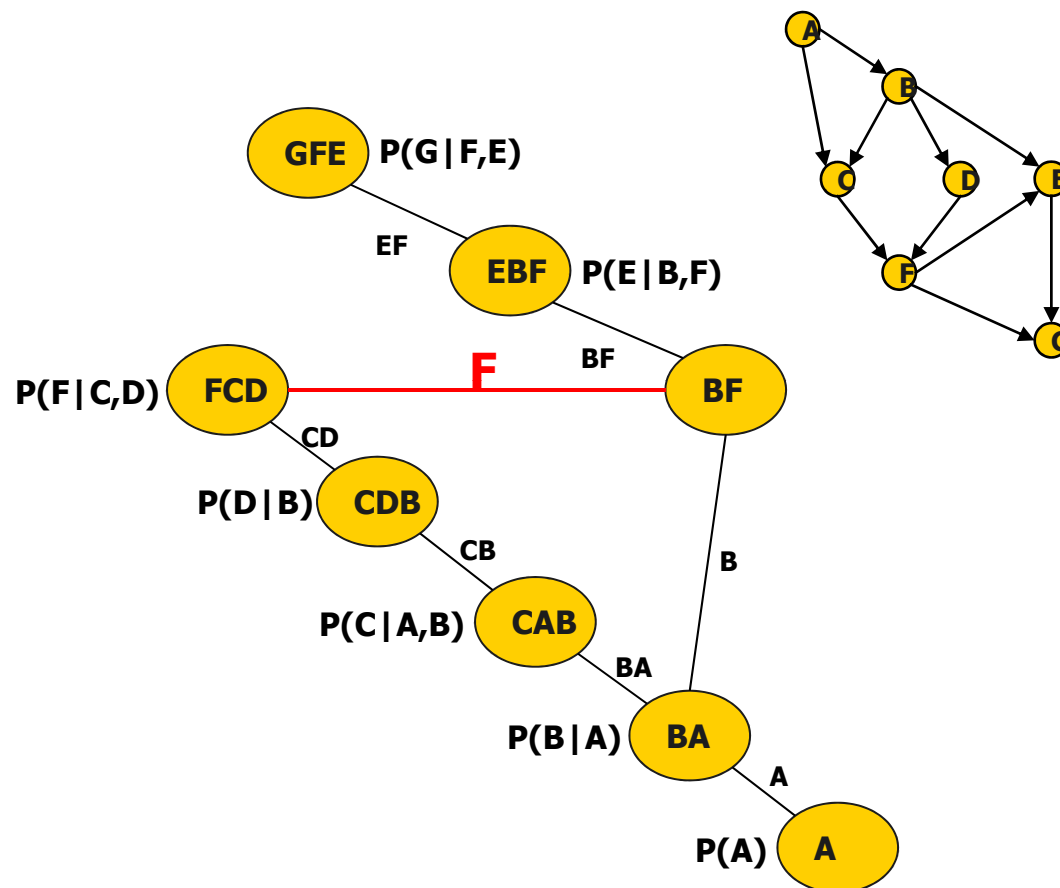
more accuracy

less complexity

Constructing Join-Graphs

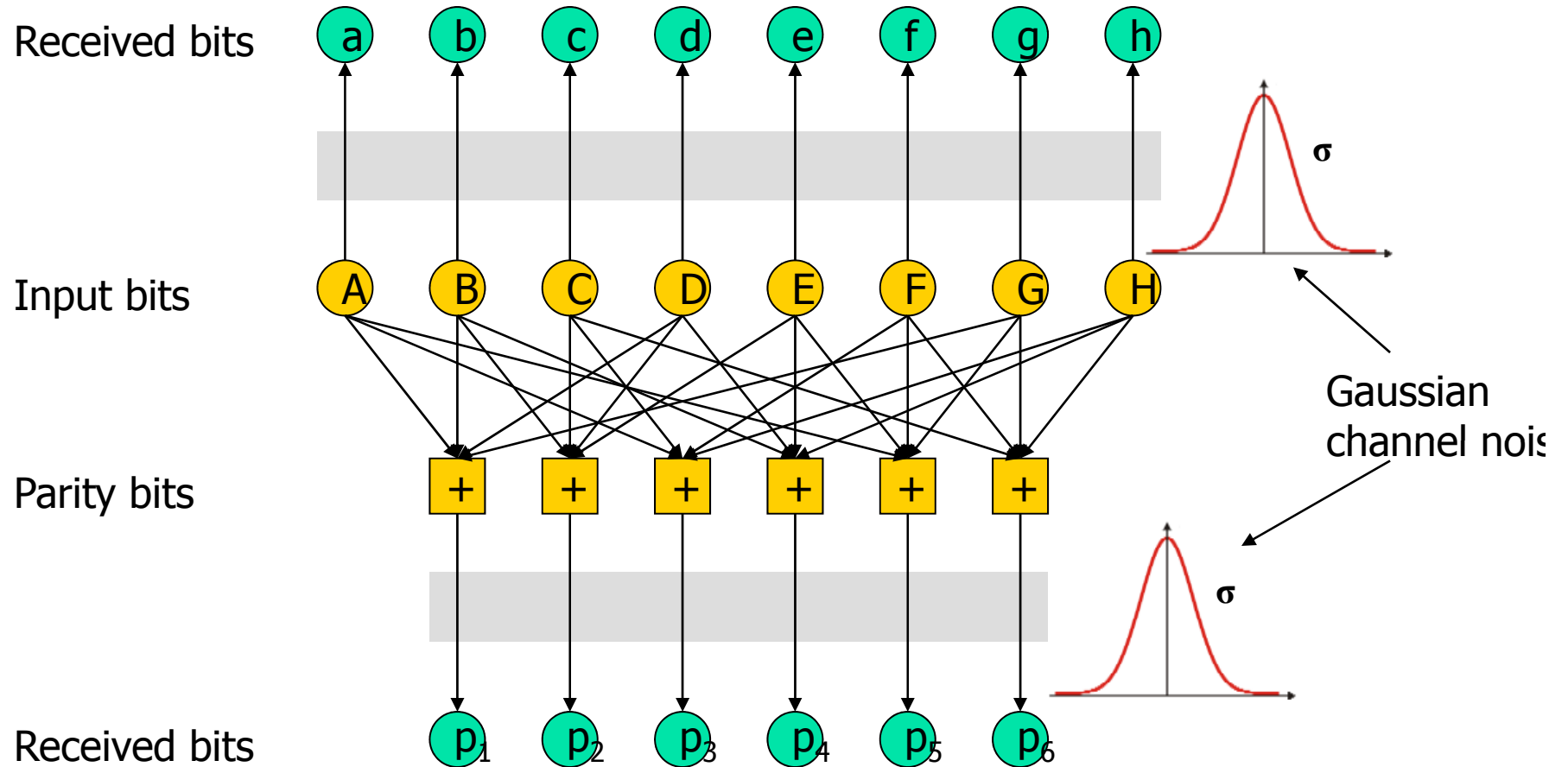


a) schematic mini-bucket(i), i=3



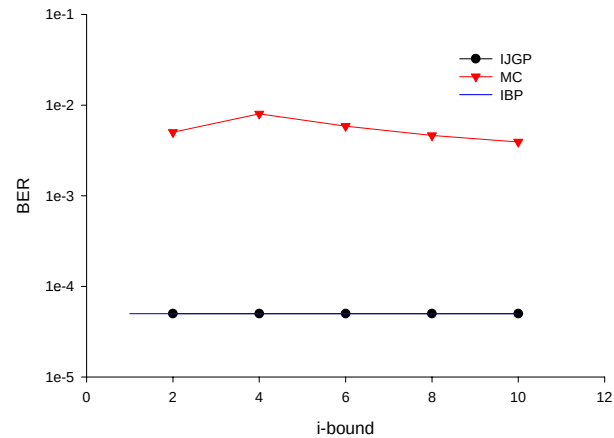
b) arc-labeled join-graph decomposition

Linear Block Codes

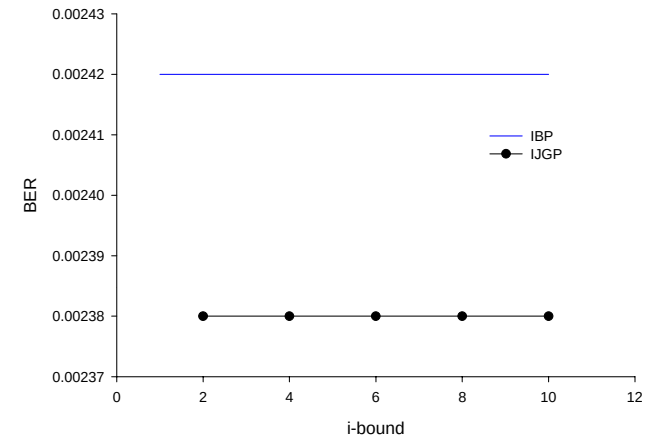


Coding Networks – Bit Error Rate

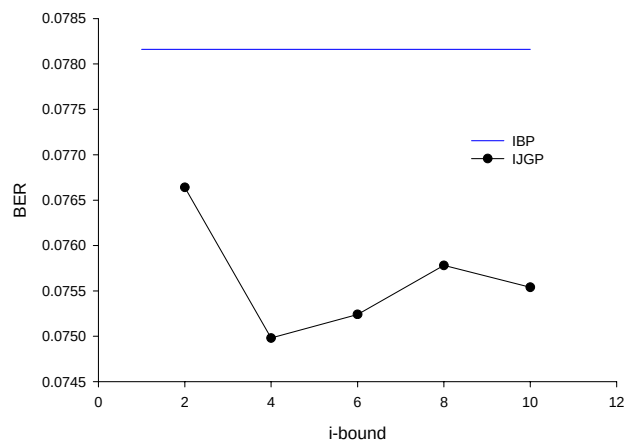
N=400, 1000 instances, 30 it, $w^*=43$, $\sigma = .22$



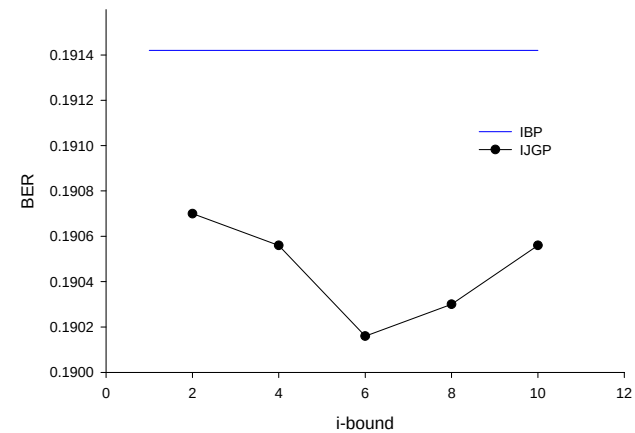
N=400, 500 instances, 30 it, $w^*=43$, $\sigma = .32$



N=400, 500 instances, 30 it, $w^*=43$, $\sigma = .51$

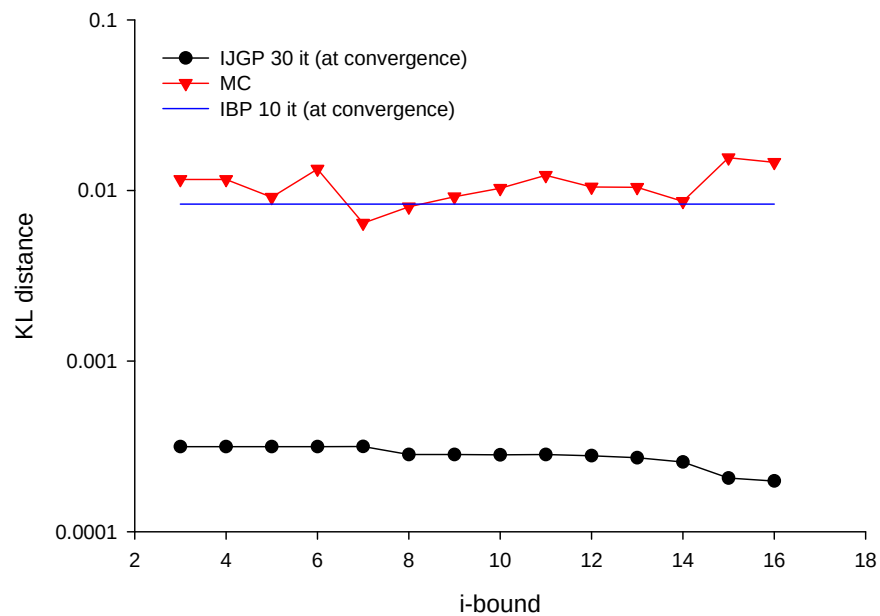


N=400, 500 instances, 30 it, $w^*=43$, $\sigma = .65$



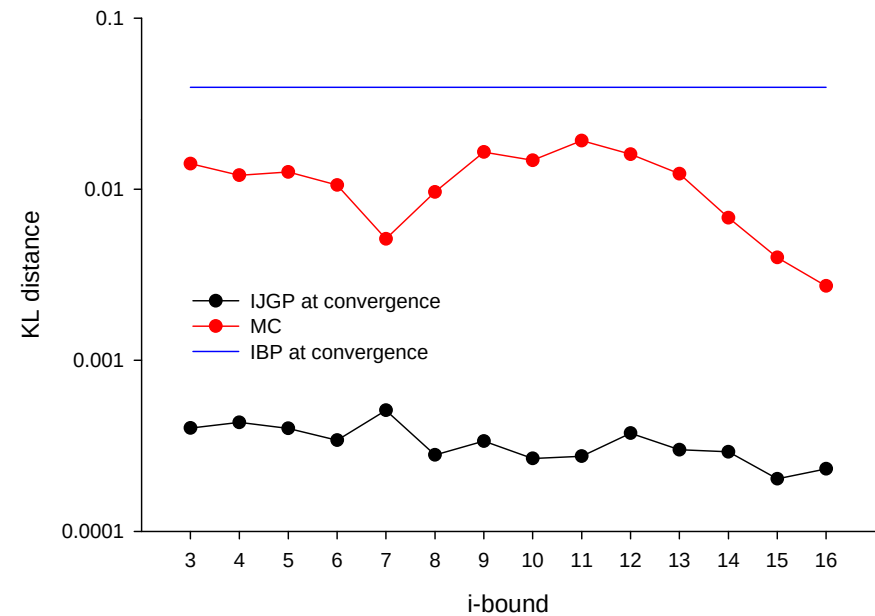
CPCS 422 – KL Distance

CPCS 422, evid=0, w*=23, 1instance



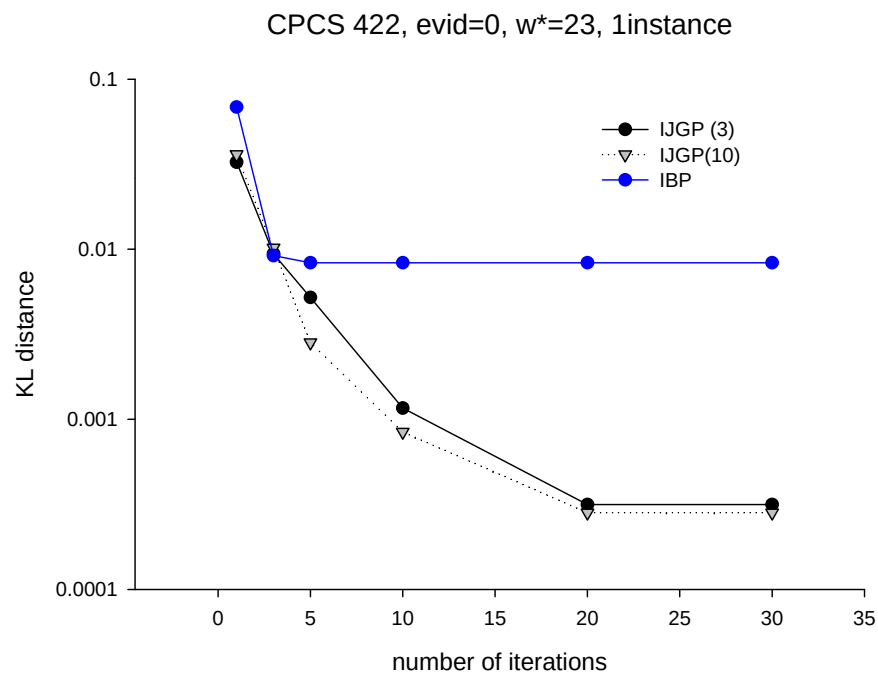
evidence=0

CPCS 422, evid=30, w*=23, 1instance

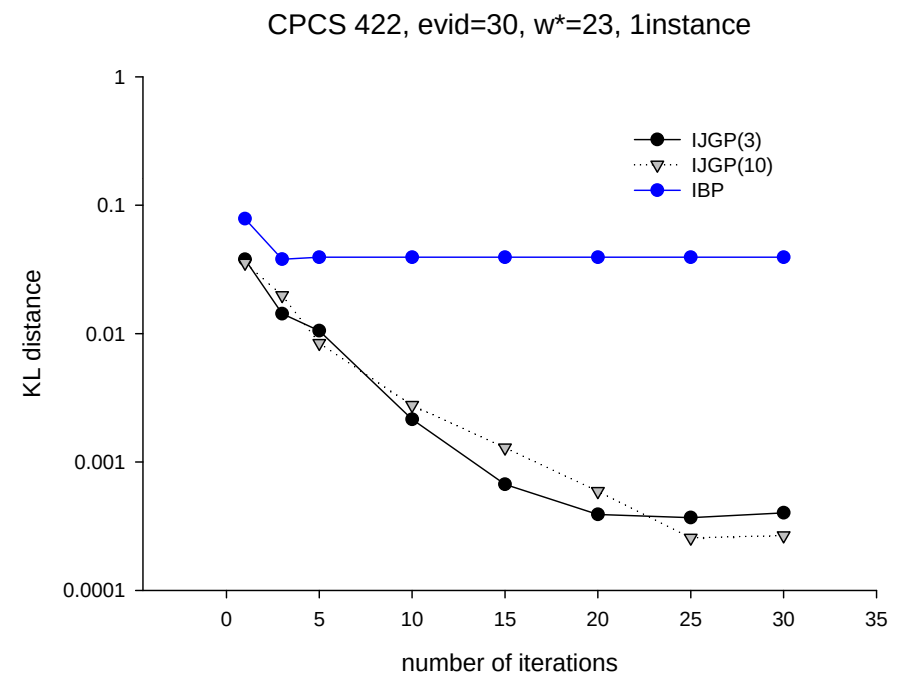


evidence=30

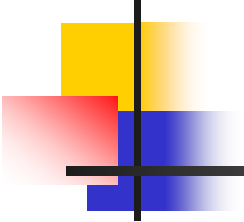
CPCS 422 – KL vs. Iterations



evidence=0



evidence=30



Inference Power of Loopy BP

- Comparison with iterative algorithms in **constraint networks**
- Zero-beliefs inconsistent assignments
 $\Leftrightarrow$
- ϵ -small beliefs – experimental study

Constraint networks

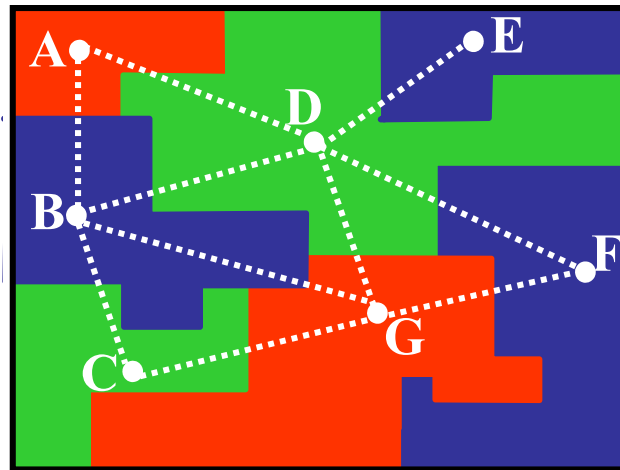
Map coloring

Variables: countries (A B C etc.)

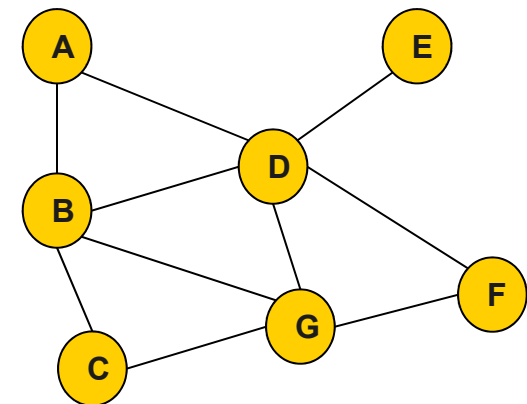
Values: colors (red green blue)

Constraints: $A \neq B$, $A \neq D$, $D \neq E$, etc.

A	B
red	green
red	yellow
green	red
green	yellow
yellow	green
yellow	red

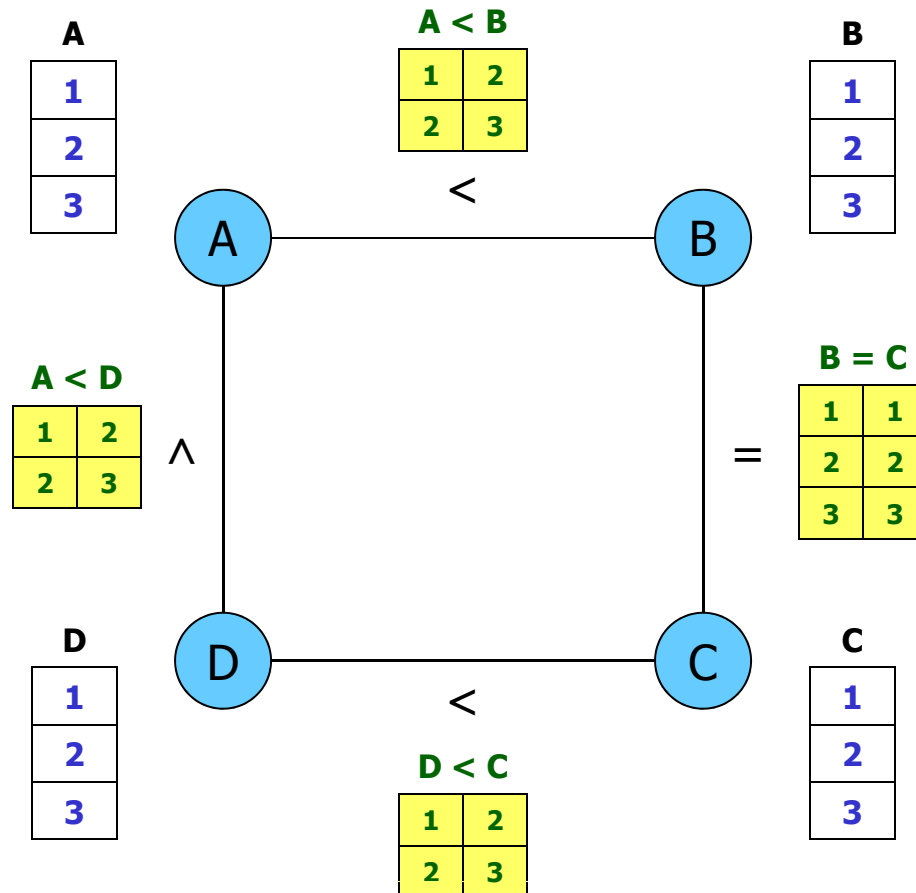


Constraint graph



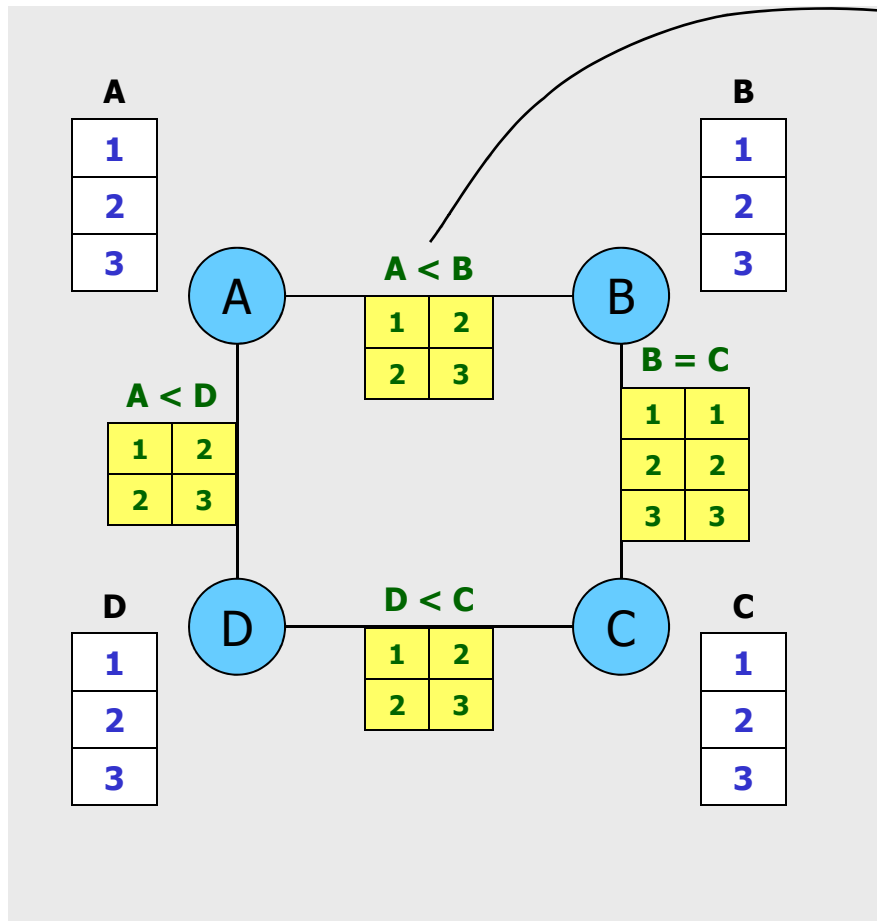
Arc-consistency

- Sound
- Incomplete
- Always converges (polynomial)

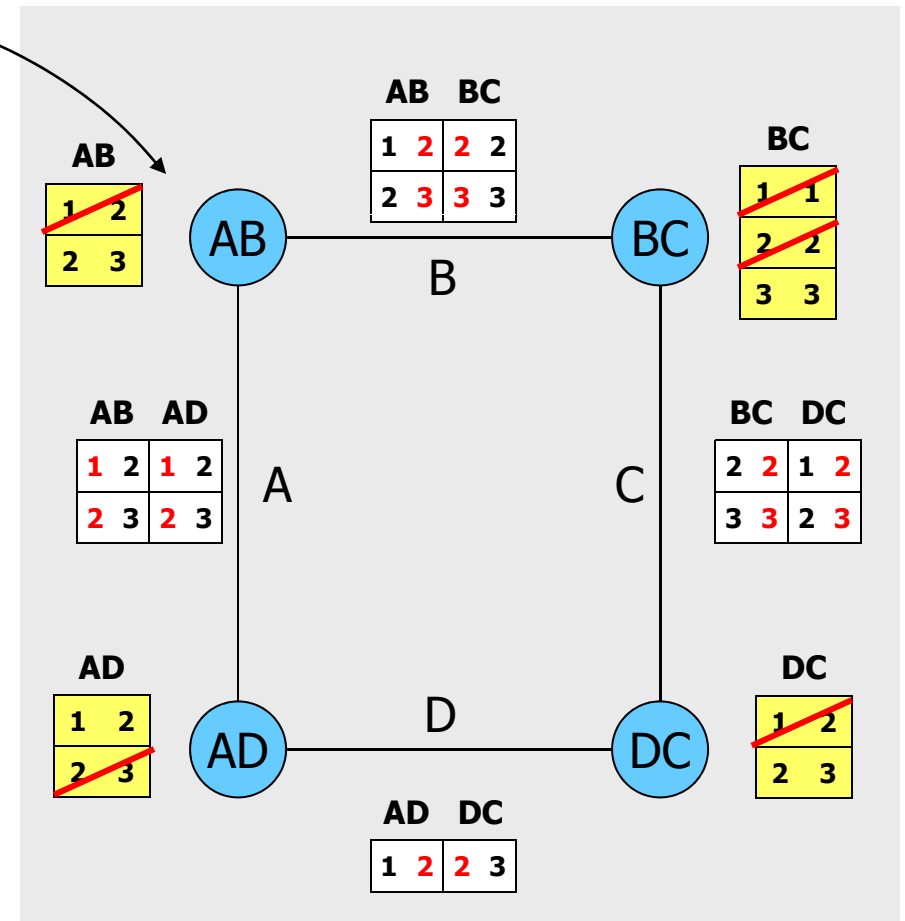


Relational Distributed Arc-Consistency

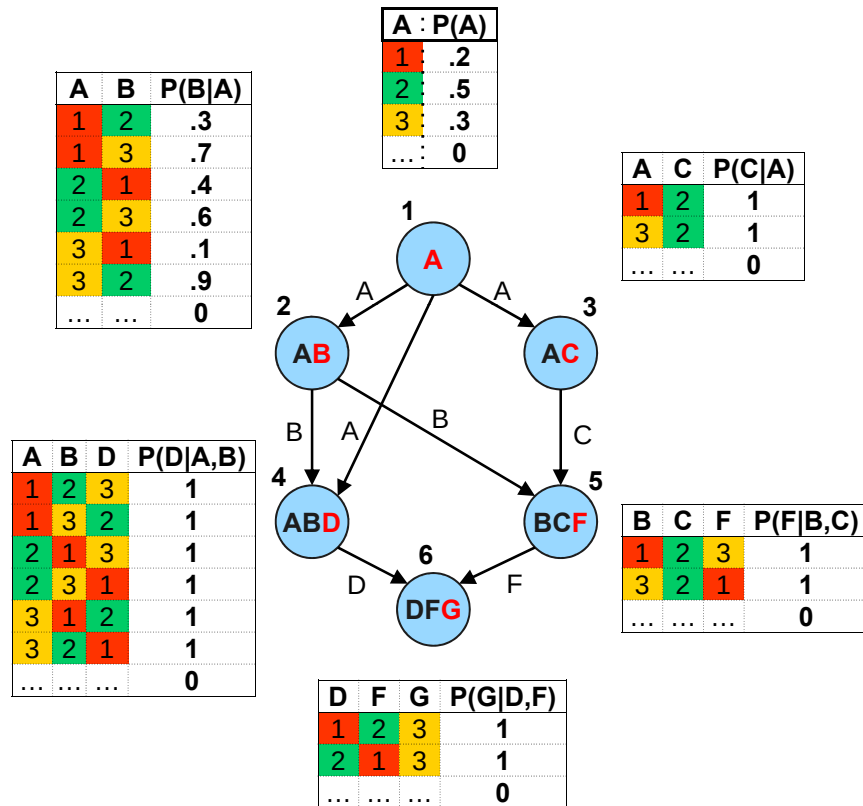
Primal



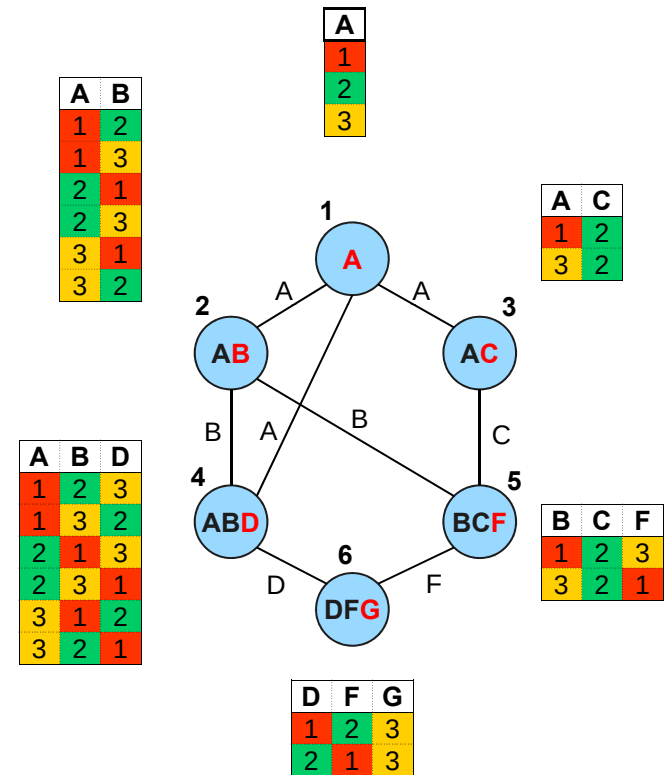
Dual



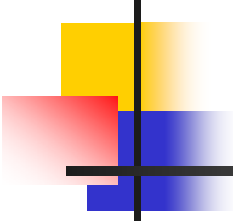
Flattening the Bayesian Network



Belief network



Flat constraint network



LBP – inference power for zero beliefs

■ Theorem:

Trace of zero beliefs of **Loopy Belief Propagation** =
Trace of invalid tuples of **arc-consistency** on flat network

■ Soundness:

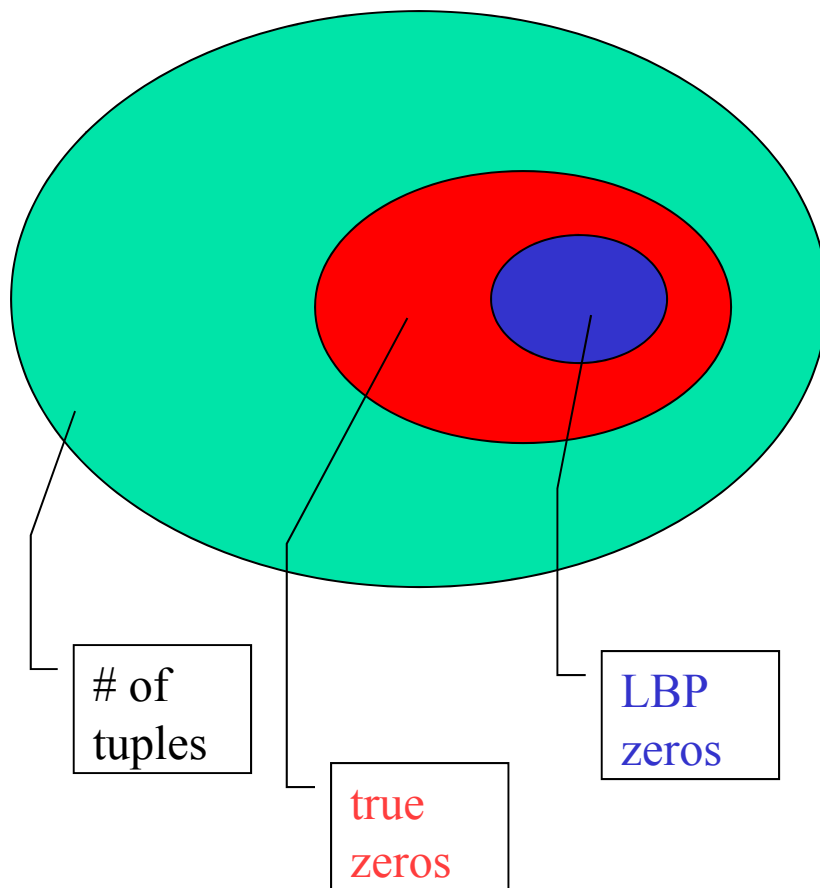
- The inference of zero beliefs by Loopy BP **converges** in a finite number of iterations
- **all the inferred zero beliefs are correct**

■ Incompleteness:

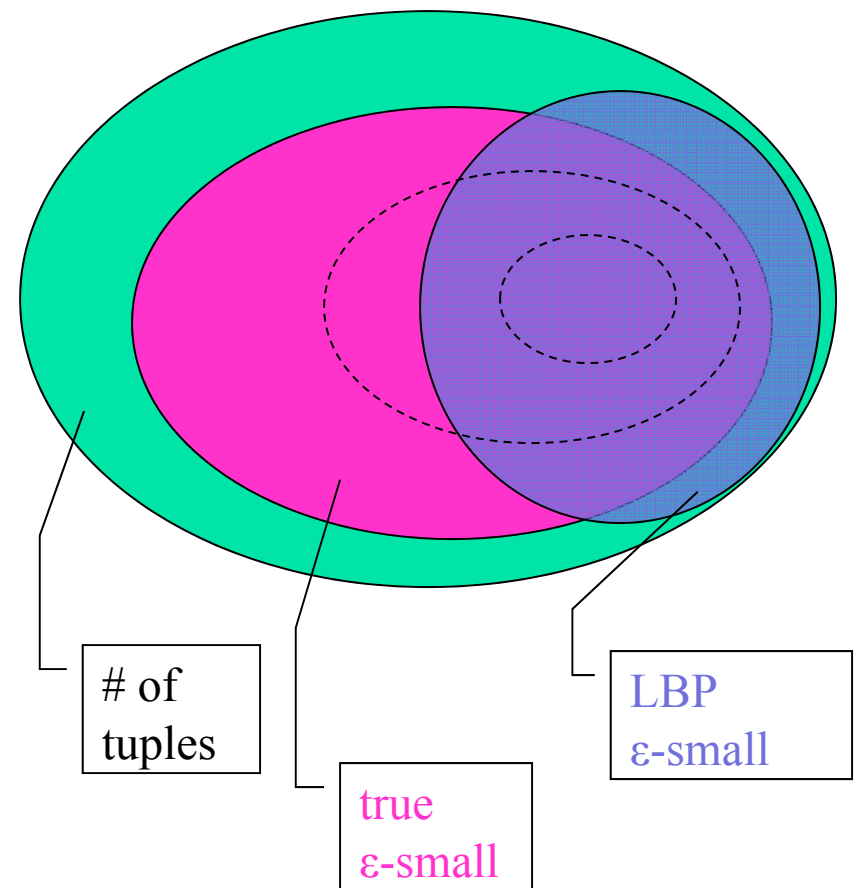
- Loopy BP may not infer all the true zero beliefs

Zero and ϵ -Small Beliefs

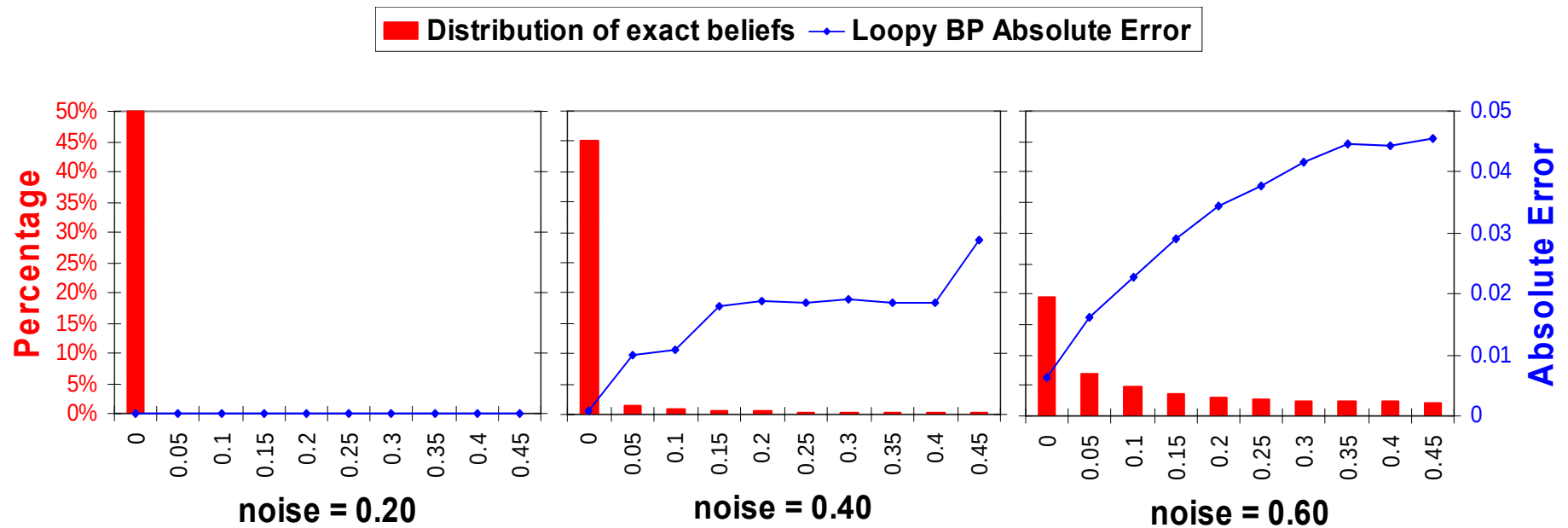
Zero beliefs



ϵ -small beliefs

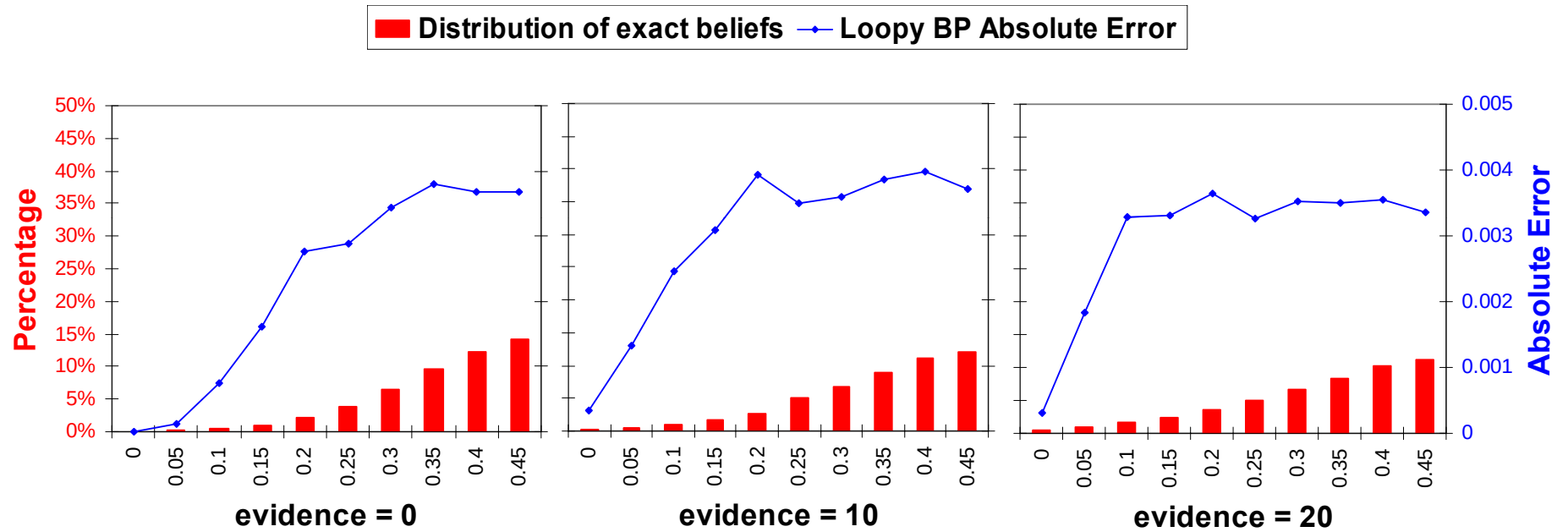


Coding Networks



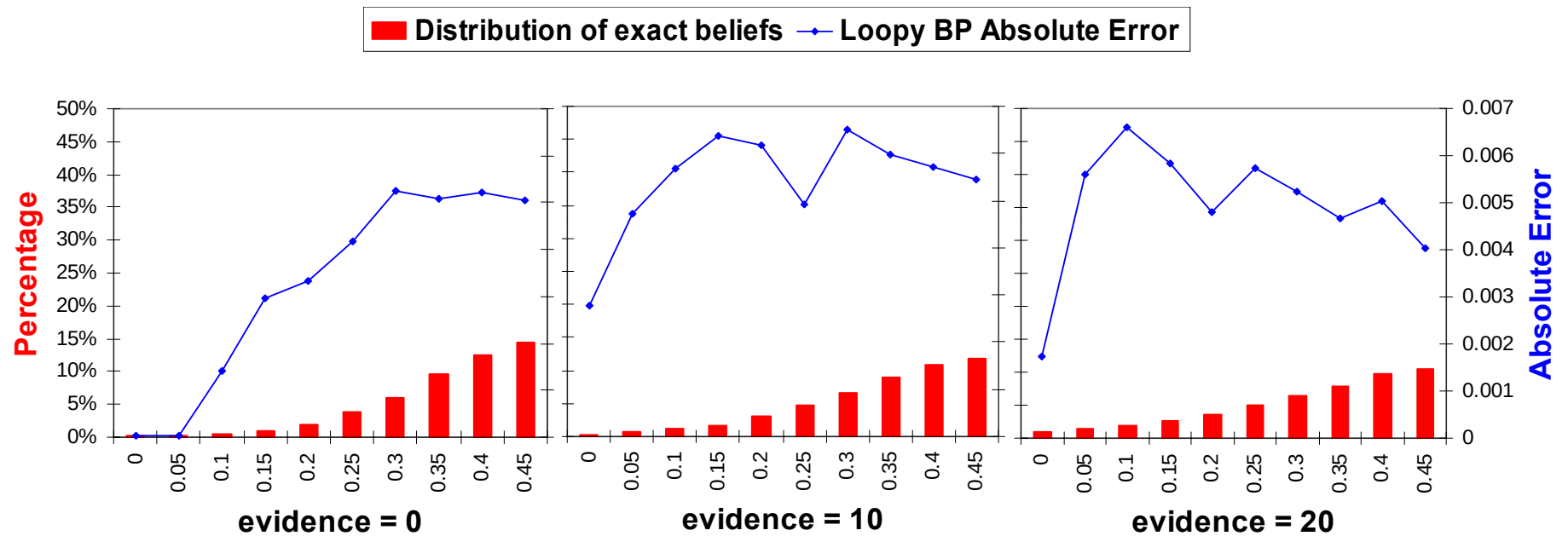
N=200, 1000 instances, treewidth=15

10x10 Grids



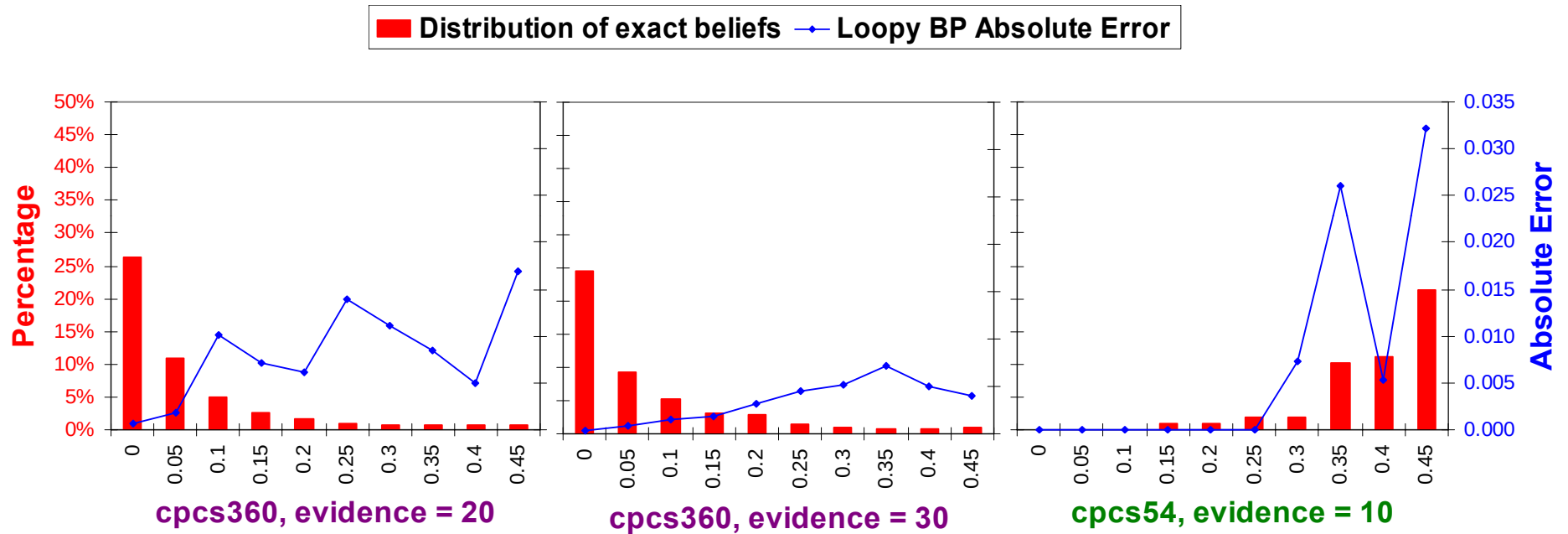
N=100, 100 instances, $w^*=15$

Random Networks



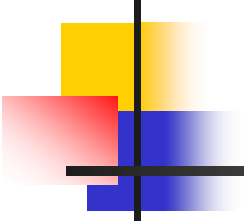
$N=80$, 100 instances, $w^*=15$

CPCS 54, CPCS360



CPCS360: 5 instances, $w^*=20$

CPCS54: 100 instances, $w^*=15$



Outline

- Introduction
- Inference
- **Search**
 - **Exact**
 - **Approximate: Sampling etc.**
- Compilation: AND/OR Decision Diagrams
- Software

Outline

■ Introduction

■ Inference

■ Search

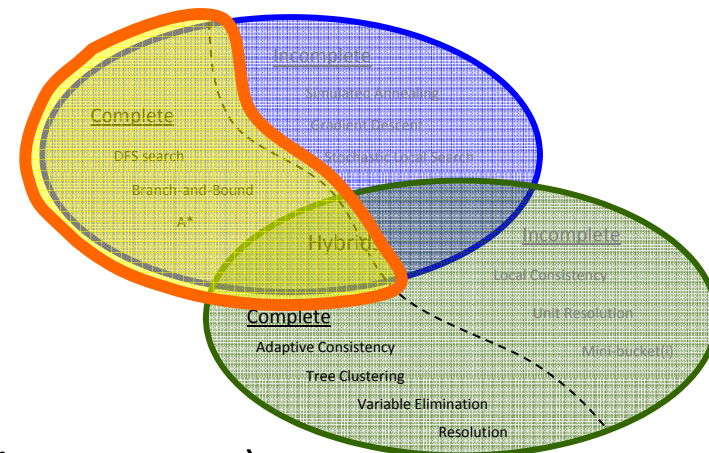
■ Exact

- AND/OR search trees (linear space)
- AND/OR Branch-and-Bound search
- AND/OR search graphs (caching)
- AND/OR search for 0-1 integer programming
- AND/OR search for multi-objective optimization

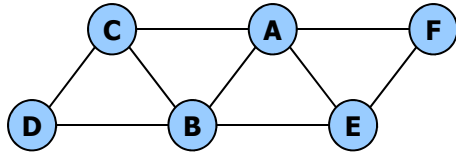
■ Approximate: Sampling etc.

■ Compilation: AND/OR Decision Diagrams

■ Software

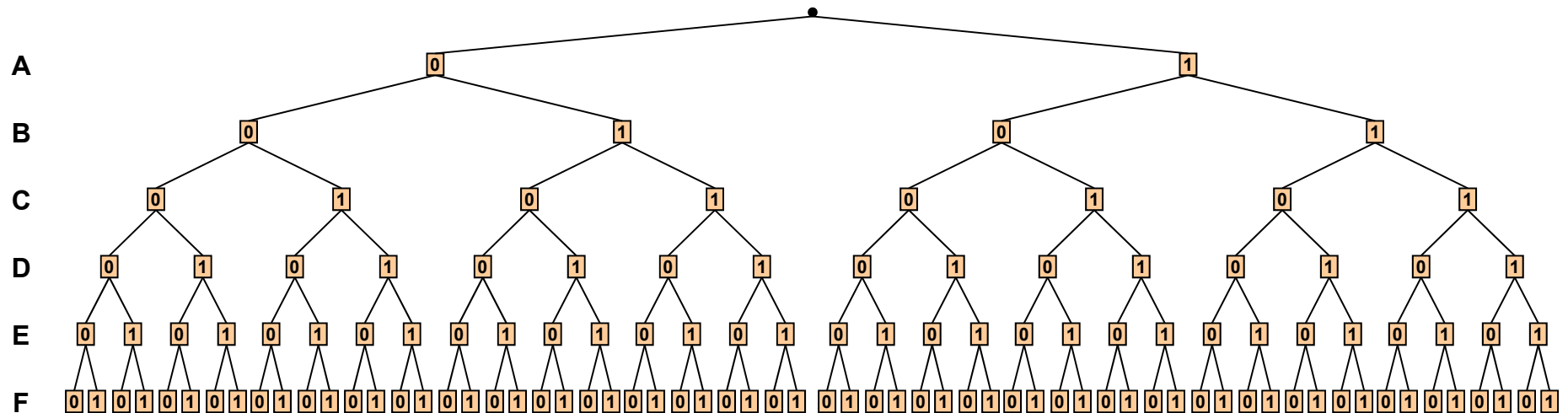


Classic OR Search Space

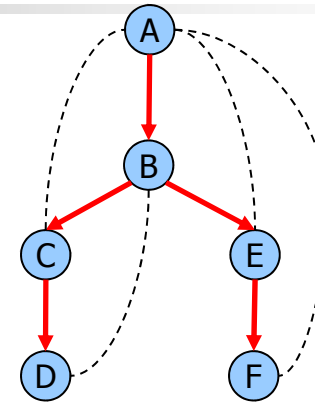
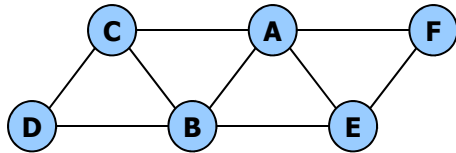


A	B	f ₁	A	C	f ₂	A	E	f ₃	A	F	f ₄	B	C	f ₅	B	D	f ₆	B	E	f ₇	C	D	f ₈	E	F	f ₉
0	0	2	0	0	3	0	0	0	0	0	2	0	0	0	0	0	4	0	0	3	0	0	1	0	0	1
0	1	0	0	1	0	0	1	3	0	1	0	0	1	1	0	1	2	0	1	2	0	1	4	0	1	0
1	0	1	1	0	0	1	0	2	1	0	0	1	0	2	1	0	1	1	0	1	1	0	0	1	0	0
1	1	4	1	1	1	1	1	0	1	1	2	1	1	4	1	1	0	1	1	0	1	1	0	1	1	2

$$f(X) = \min_x \sum_{i=1}^9 f_i(X)$$



The AND/OR Search Tree



Pseudo tree (Freuder & Quinn85)

OR

AND

OR

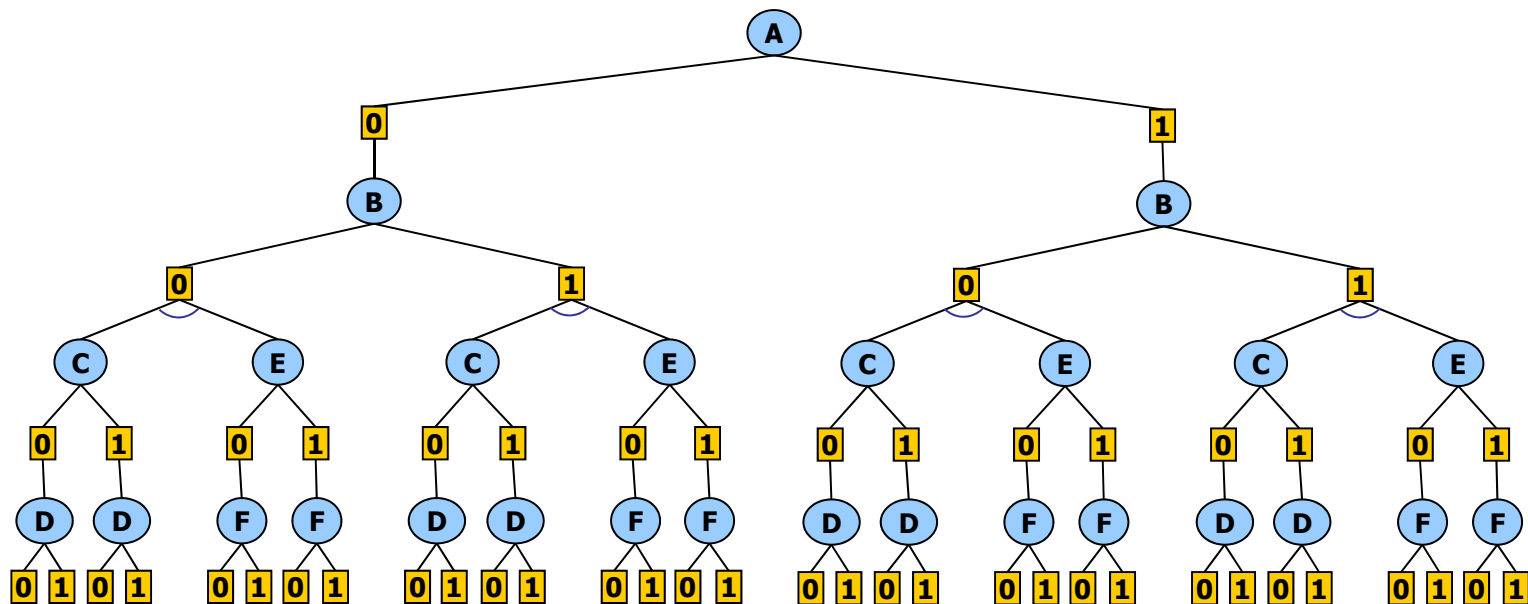
AND

OR

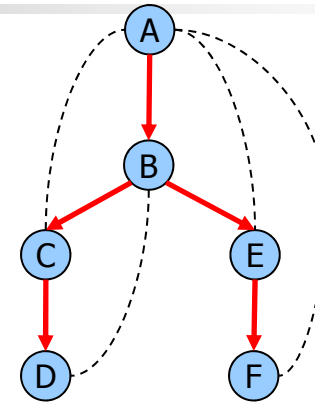
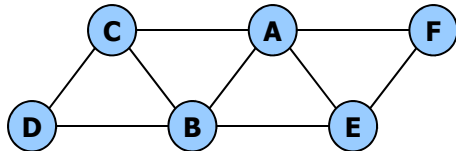
AND

OR

AND

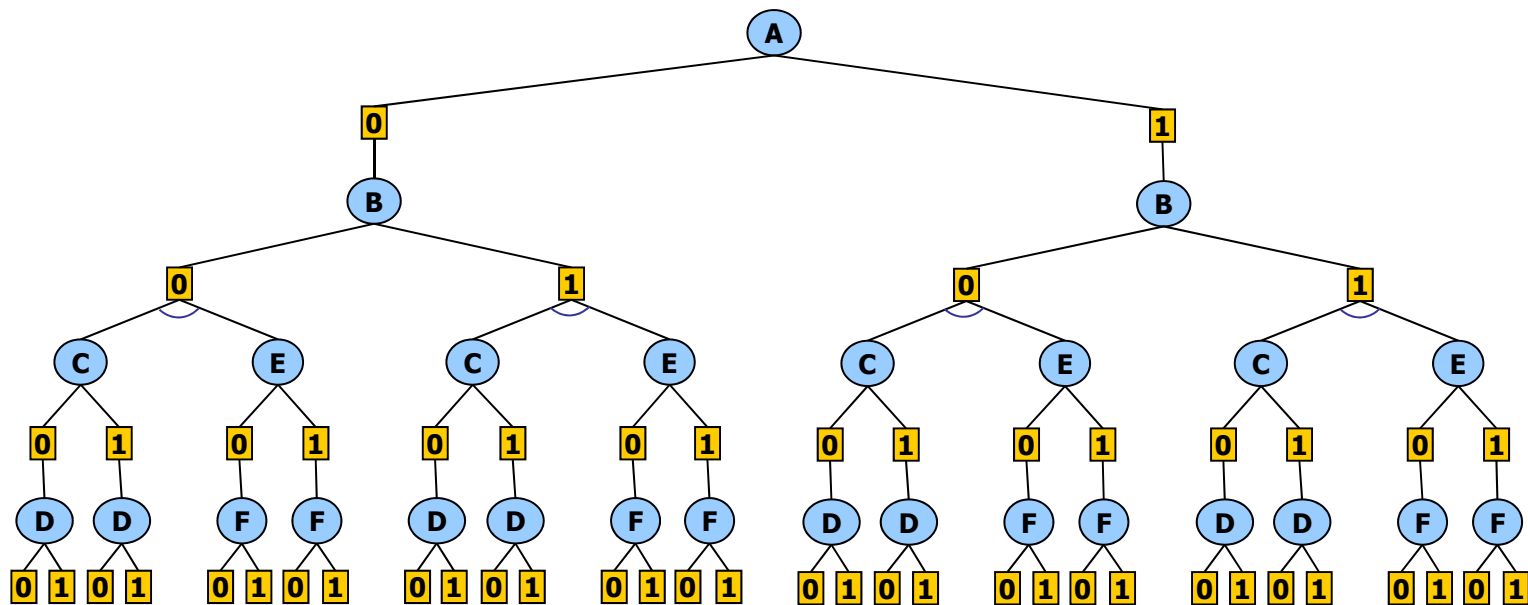


The AND/OR Search Tree



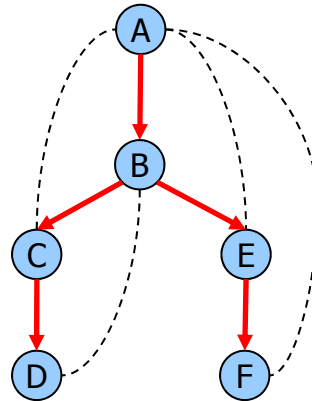
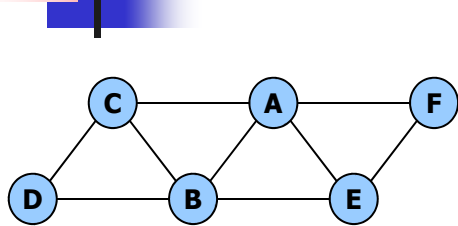
Pseudo tree

OR
AND
OR
AND
OR
AND



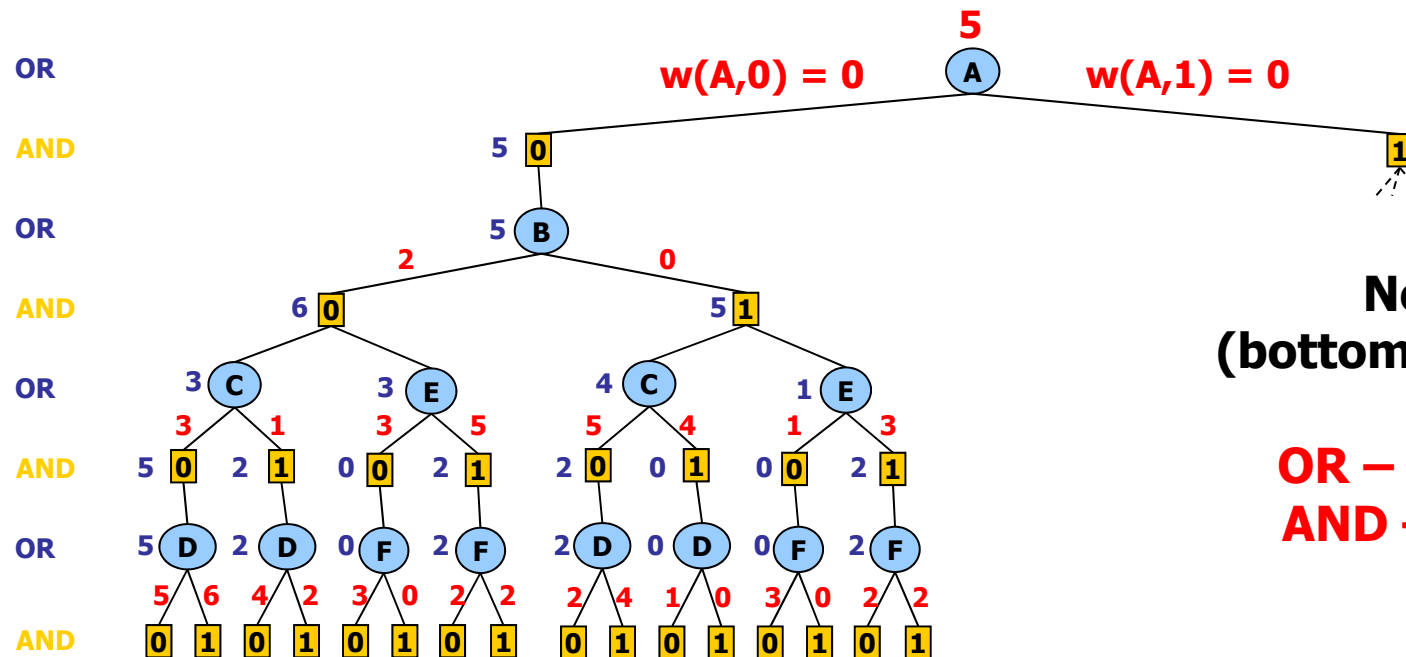
A solution subtree is (A=0, B=1, C=0, D=0, E=1, F=1)

Weighted AND/OR Search Tree



A	B	f ₁	A	C	f ₂	A	E	f ₃	A	F	f ₄	B	C	f ₅	B	D	f ₆	B	E	f ₇	C	D	f ₈	E	F	f ₉
0	0	2	0	0	3	0	0	0	0	0	2	0	0	0	0	0	4	0	0	3	0	0	1	0	0	1
0	1	0	0	1	0	0	1	3	0	1	0	0	1	1	0	1	2	0	1	2	0	1	4	0	1	0
1	0	1	1	0	0	1	0	2	1	0	0	1	0	2	1	0	1	1	0	1	1	0	0	1	0	0
1	1	4	1	1	1	1	1	0	1	1	2	1	1	4	1	1	0	1	1	0	1	1	0	1	1	2

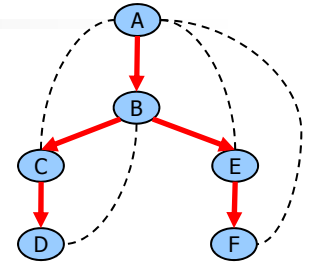
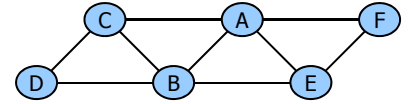
$$f(X) = \min_x \sum_{i=1}^9 f_i(X)$$



Node Value
(bottom-up evaluation)

OR – minimization
AND – summation

AND/OR vs. OR Spaces



OR

AND

OR

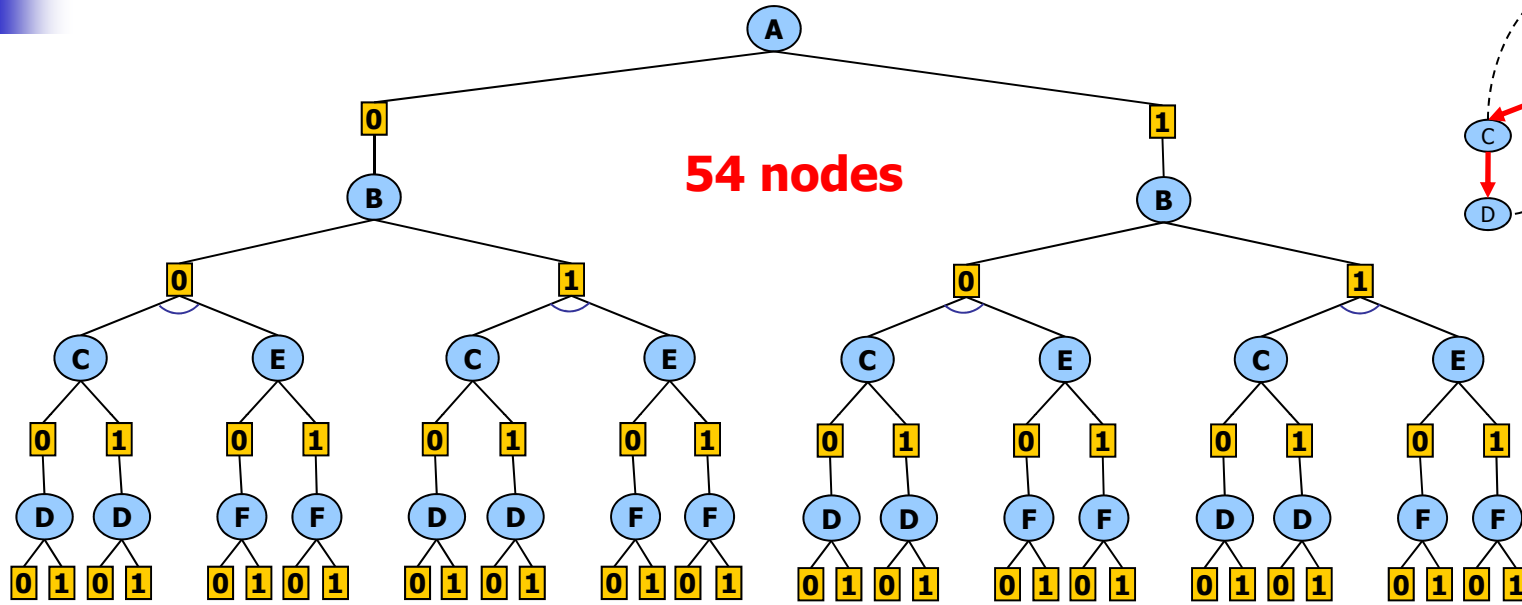
AND

OR

AND

OR

AND



A

B

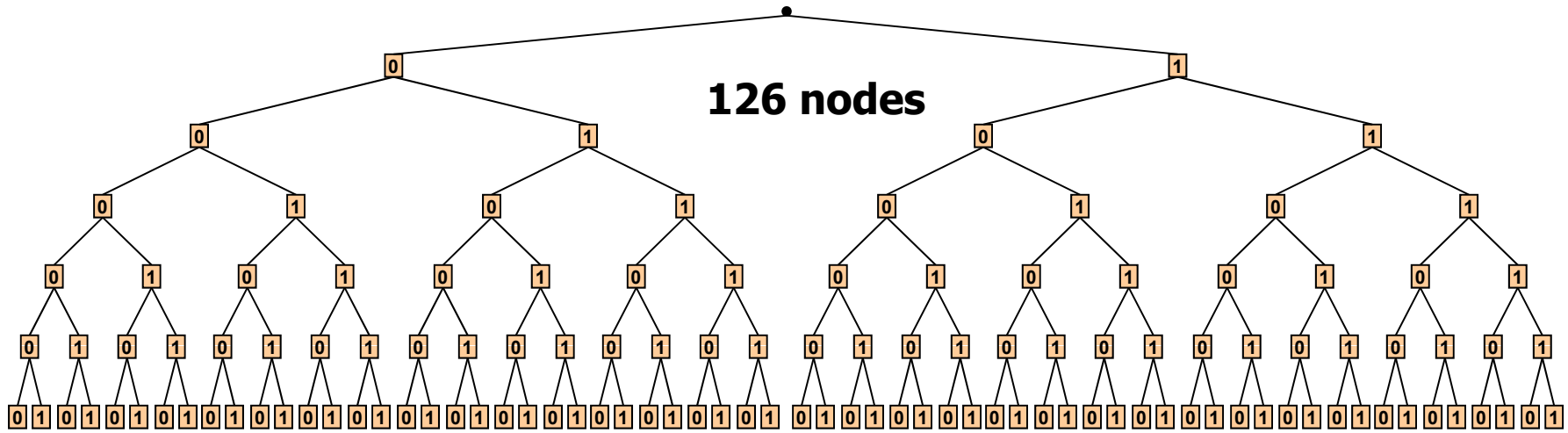
C

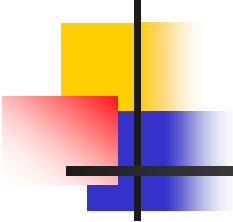
D

E

F

126 nodes

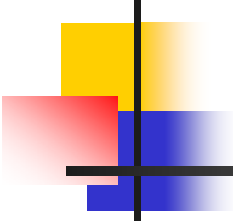




AND/OR vs. OR Spaces

width	depth	OR space		AND/OR space		
		Time (sec.)	Nodes	Time (sec.)	AND nodes	OR nodes
5	10	3.15	2,097,150	0.03	10,494	5,247
4	9	3.13	2,097,150	0.01	5,102	2,551
5	10	3.12	2,097,150	0.03	8,926	4,463
4	10	3.12	2,097,150	0.02	7,806	3,903
5	13	3.11	2,097,150	0.10	36,510	18,255

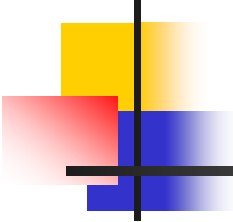
Random graphs with 20 nodes, 20 edges and 2 values per node



Complexity of AND/OR Tree Search

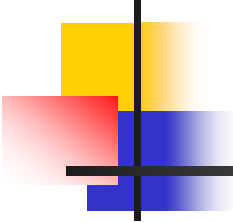
	AND/OR tree	OR tree
Space	$O(n)$	$O(n)$
Time	$O(n d^t)$ $O(n d^{w^* \log n})$ (Freuder & Quinn85), (Collin, Dechter & Katz91), (Bayardo & Miranker95), (Darwiche01)	$O(d^n)$

d = domain size
 t = depth of pseudo-tree
 n = number of variables
 w^* = treewidth



Constructing Pseudo Trees

- AND/OR search algorithms are influenced by the **quality** of the pseudo tree
- Finding the minimal induced width / depth pseudo tree is NP-hard
- Heuristics
 - **Min-Fill** (min induced width)
 - **Hypergraph partitioning** (min depth)



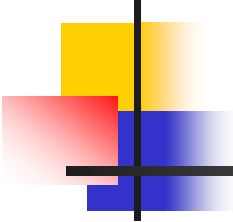
Constructing Pseudo Trees

■ **Min-Fill** (Kjaerulff90)

- Depth-first traversal of the induced graph obtained along the **min-fill** elimination order
- Variables ordered according to the smallest “fill-set”

■ **Hypergraph Partitioning** (Karypis &Kumar00)

- Functions are vertices in the hypergraph and variables are hyperedges
- Recursive decomposition of the hypergraph while minimizing the separator size at each step
- Using state-of-the-art software package **hMeTiS**



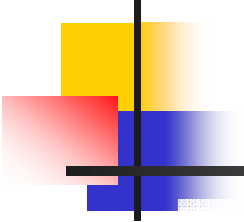
Quality of the Pseudo Trees

Network	hypergraph		min-fill	
	width	depth	width	depth
barley	7	13	7	23
diabetes	7	16	4	77
link	21	40	15	53
mildew	5	9	4	13
munin1	12	17	12	29
munin2	9	16	9	32
munin3	9	15	9	30
munin4	9	18	9	30
water	11	16	10	15
pigs	11	20	11	26

Bayesian Networks Repository

Network	hypergraph		min-fill	
	width	depth	width	depth
spot5	47	152	39	204
spot28	108	138	79	199
spot29	16	23	14	42
spot42	36	48	33	87
spot54	12	16	11	33
spot404	19	26	19	42
spot408	47	52	35	97
spot503	11	20	9	39
spot505	29	42	23	74
spot507	70	122	59	160

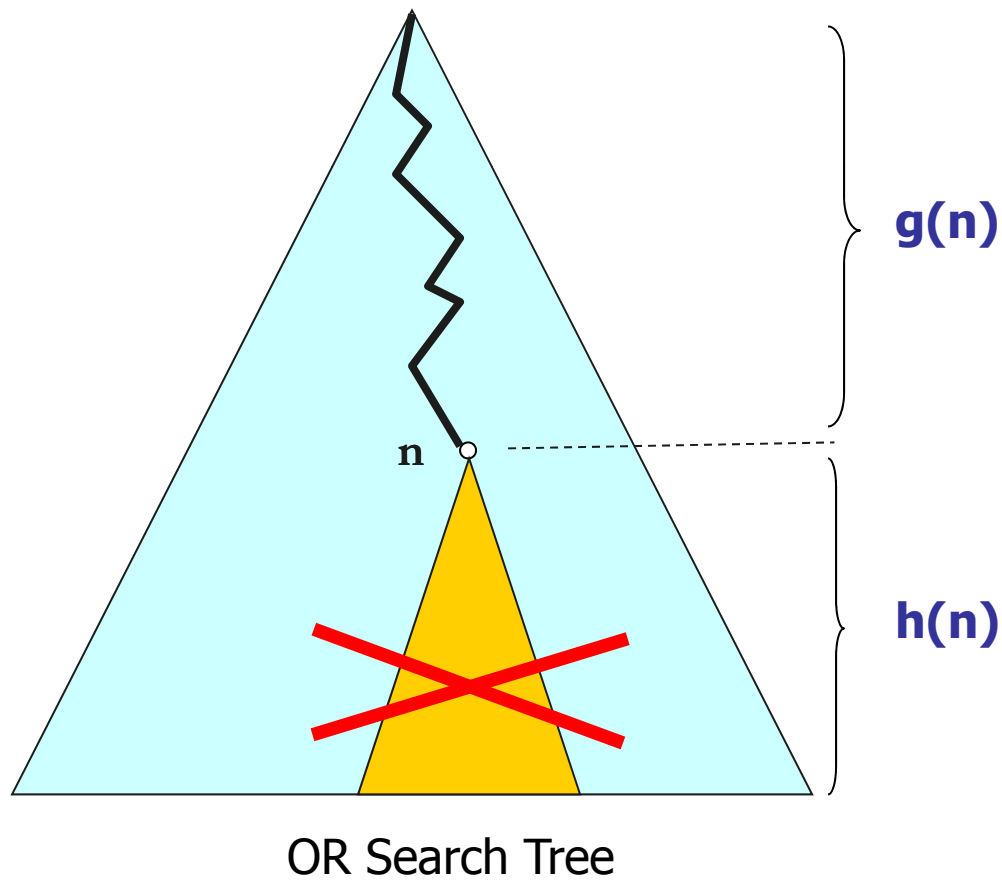
SPOT5 Benchmarks



Outline

- Introduction
- Inference
- **Search**
 - **Exact**
 - AND/OR search trees
 - **AND/OR Branch-and-Bound search**
 - Lower bounding heuristics
 - Dynamic variable orderings
 - AND/OR search graphs (caching)
 - AND/OR search for 0-1 integer programming
 - AND/OR search for multi-objective optimization
 - Approximate: Sampling etc.
- Compilation: AND/OR Decision Diagrams
- Software

Classic Branch-and-Bound Search



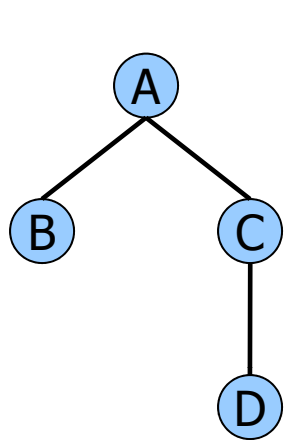
Upper Bound **UB**

Lower Bound **LB**

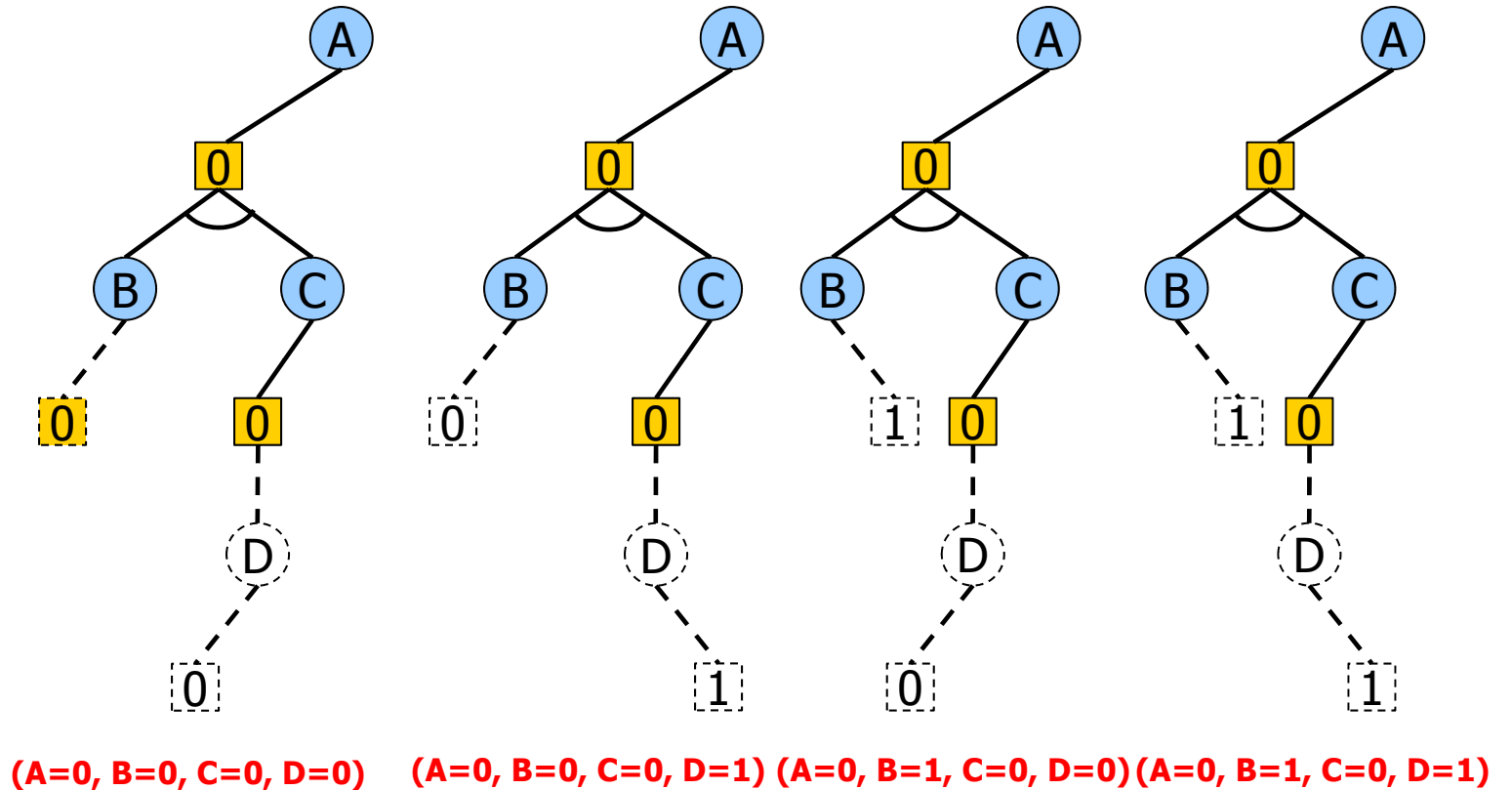
$$\mathbf{LB(n) = g(n) + h(n)}$$

Prune if $LB(n) \geq UB$

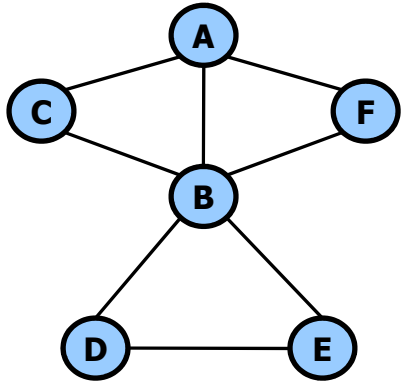
Partial Solution Tree



Pseudo tree



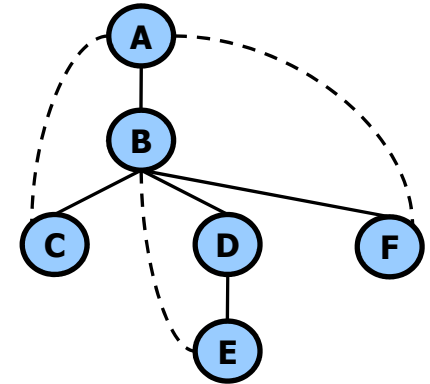
Extension(T') – solution trees that extend T'



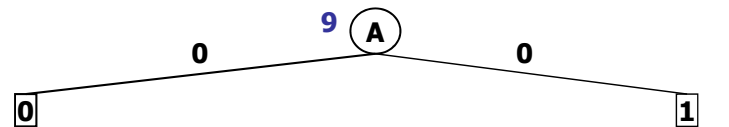
A	B	C	$f_1(ABC)$
0	0	0	2
0	0	1	5
0	1	0	3
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	2

A	B	F	$f_2(ABF)$
0	0	0	3
0	0	1	5
0	1	0	1
0	1	1	4
1	0	0	6
1	0	1	5
1	1	0	6
1	1	1	5

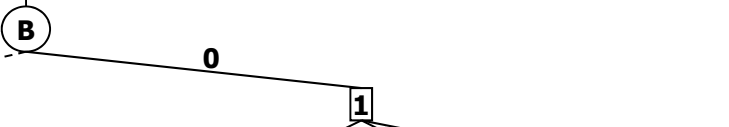
B	D	E	$f_3(\text{BDE})$
0	0	0	6
0	0	1	4
0	1	0	8
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	4



OR

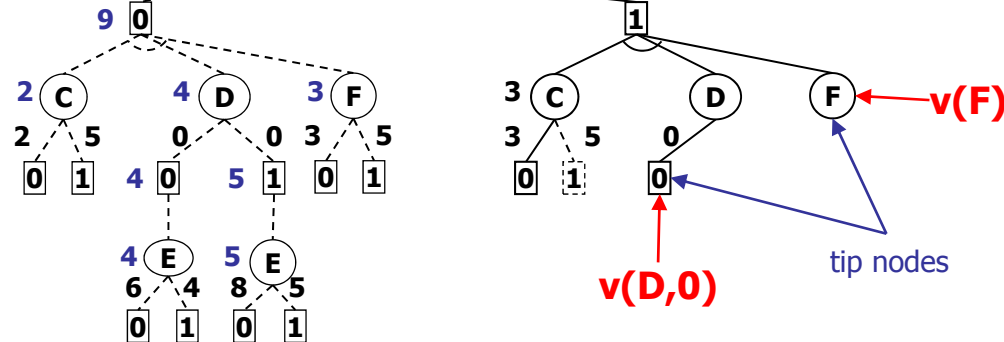


AND



OR

AND



OR

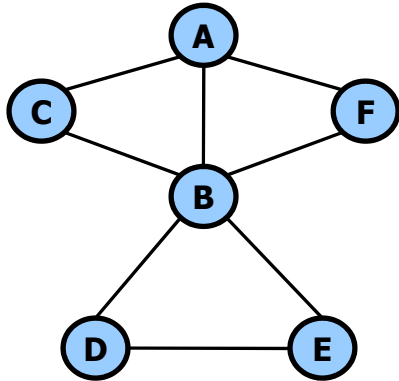
AND

OR

AND

$$f^*(T') = w(A,0) + w(B,1) + w(C,0) + w(D,0) + v(D,0) + v(F)$$

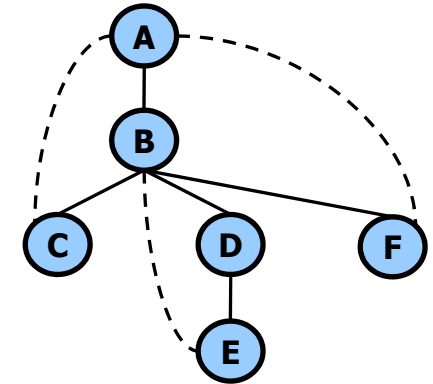
Heuristic Evaluation Function



A	B	C	$f_1(ABC)$
0	0	0	2
0	0	1	5
0	1	0	3
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	2

A	B	F	$f_2(ABF)$
0	0	0	3
0	0	1	5
0	1	0	1
0	1	1	4
1	0	0	6
1	0	1	5
1	1	0	6
1	1	1	5

B	D	E	$f_3(BDE)$
0	0	0	6
0	0	1	4
0	1	0	8
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	4



OR

AND

OR

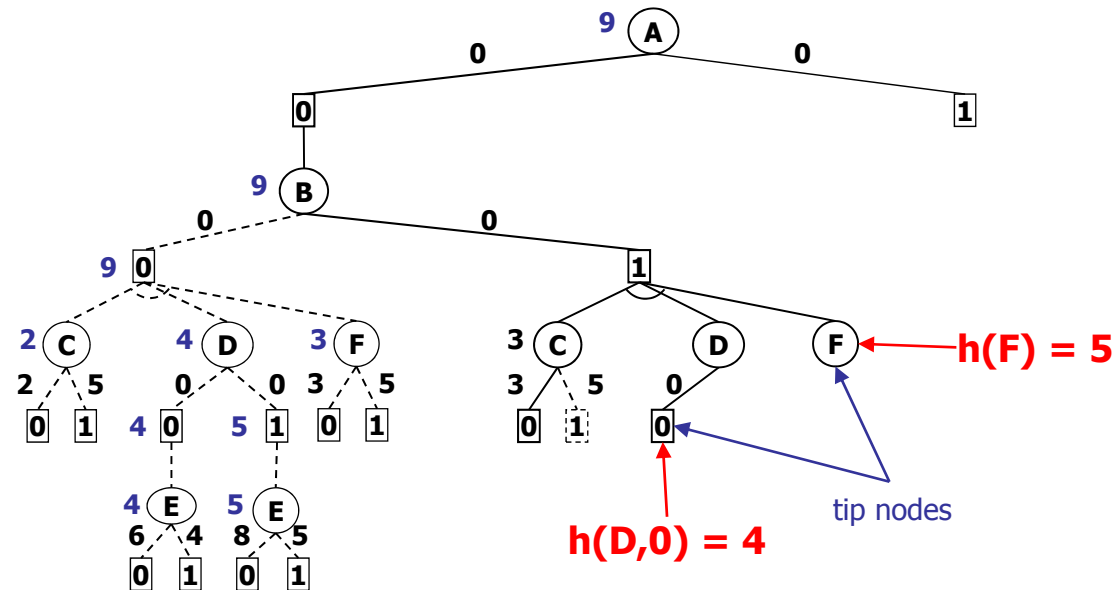
AND

OR

AND

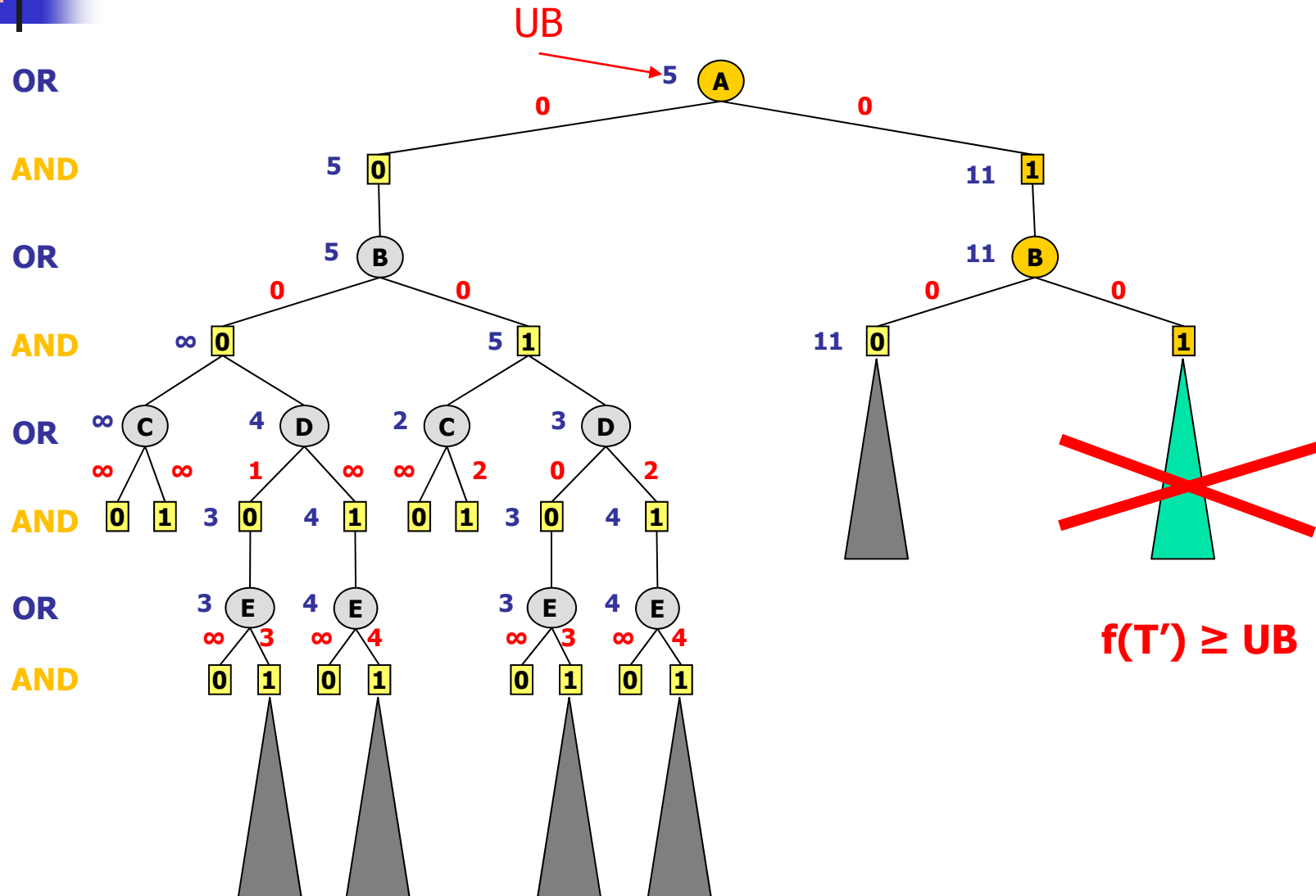
OR

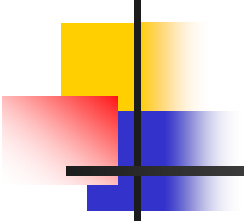
AND



$$f(T') = w(A,0) + w(B,1) + w(C,0) + w(D,0) + h(D,0) + h(F) = 12 \leq f^*(T')$$

AND/OR Branch-and-Bound Search

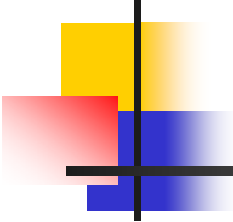




AND/OR Branch-and-Bound Search (AOBB)

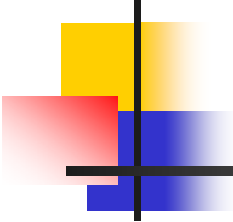
(Marinescu & Dechter, IJCAI'05)

- Associate each node n with a heuristic lower bound $h(n)$ on $v(n)$
- EXPAND (top-down)
 - Evaluate $f(T')$ and prune search if $f(T') \geq UB$
 - Expand the tip node n
- PROPAGATE (bottom-up)
 - Update value of the parent p of n
 - OR nodes: minimization
 - AND nodes: summation



Heuristics for AND/OR Branch-and-Bound

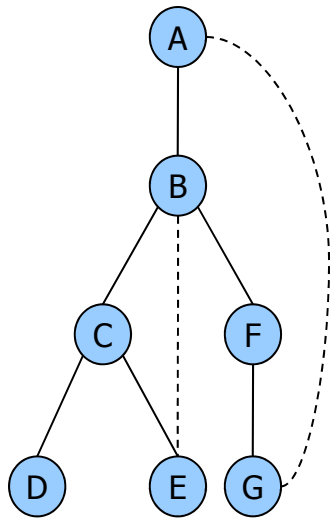
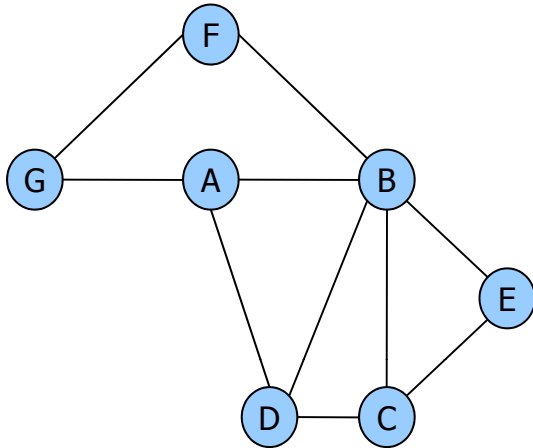
- In the AND/OR search space $h(n)$ can be computed using any heuristic. We used:
 - Static Mini-Bucket heuristics
(Kask & Dechter, AIJ'01), (Marinescu & Dechter, IJCAI'05)
 - Dynamic Mini-Bucket heuristics
(Marinescu & Dechter, IJCAI'05)
 - Maintaining local consistency
(Larrosa & Schiex, AAAI'03), (de Givry et al., IJCAI'05)
 - LP relaxations
(Nemhauser & Wolsey, 1998)



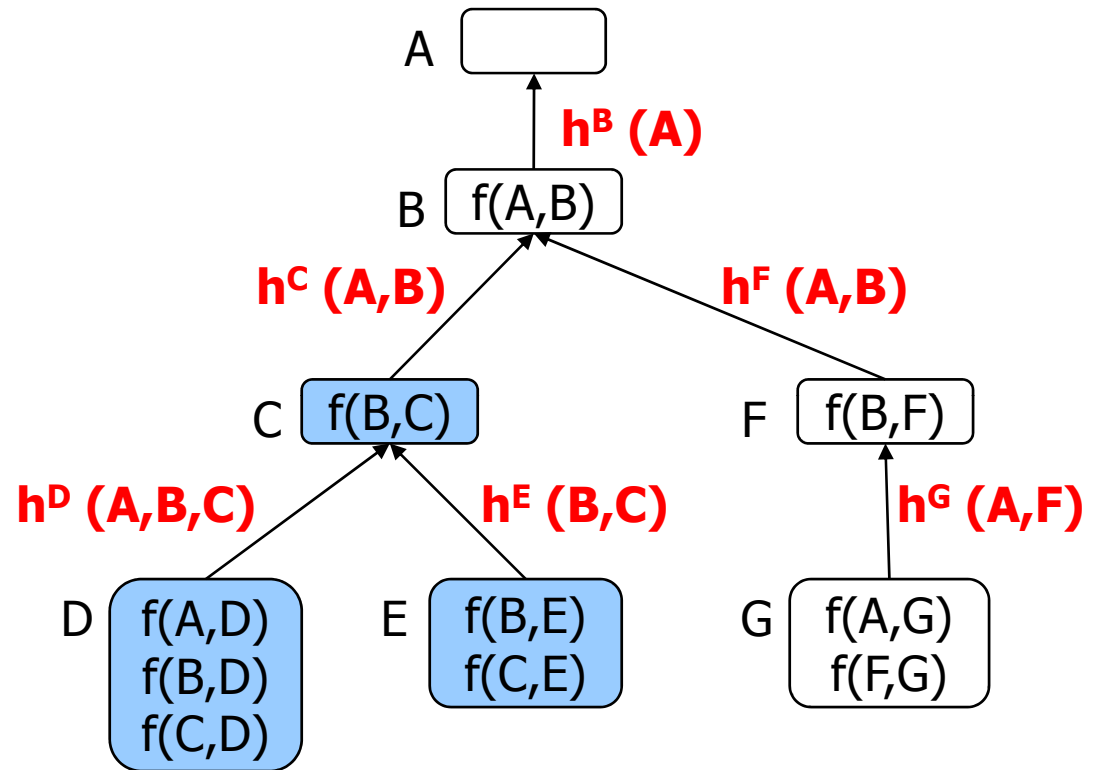
Mini-Bucket Heuristics

- | | |
|----------------------------|-----------------------------|
| ■ Static Mini-Buckets | ■ Dynamic Mini-Buckets |
| ■ Pre-compiled | ■ Computed dynamically |
| ■ Reduced overhead | ■ Higher overhead |
| ■ Less accurate | ■ High accuracy |
| ■ Static variable ordering | ■ Dynamic variable ordering |

Bucket Elimination

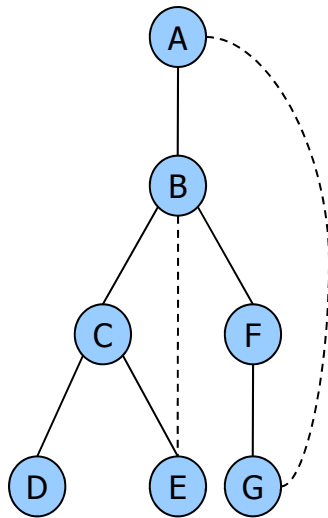
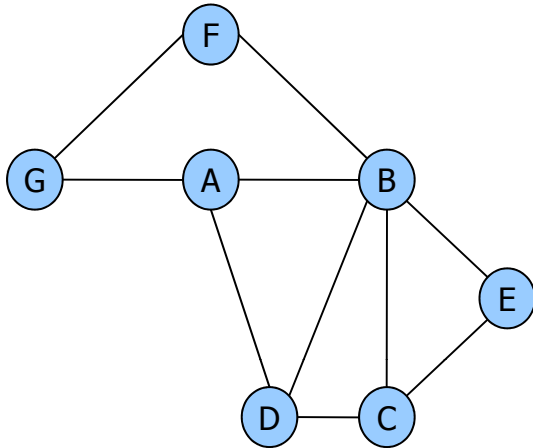


Ordering: (A, B, C, D, E, F, G)

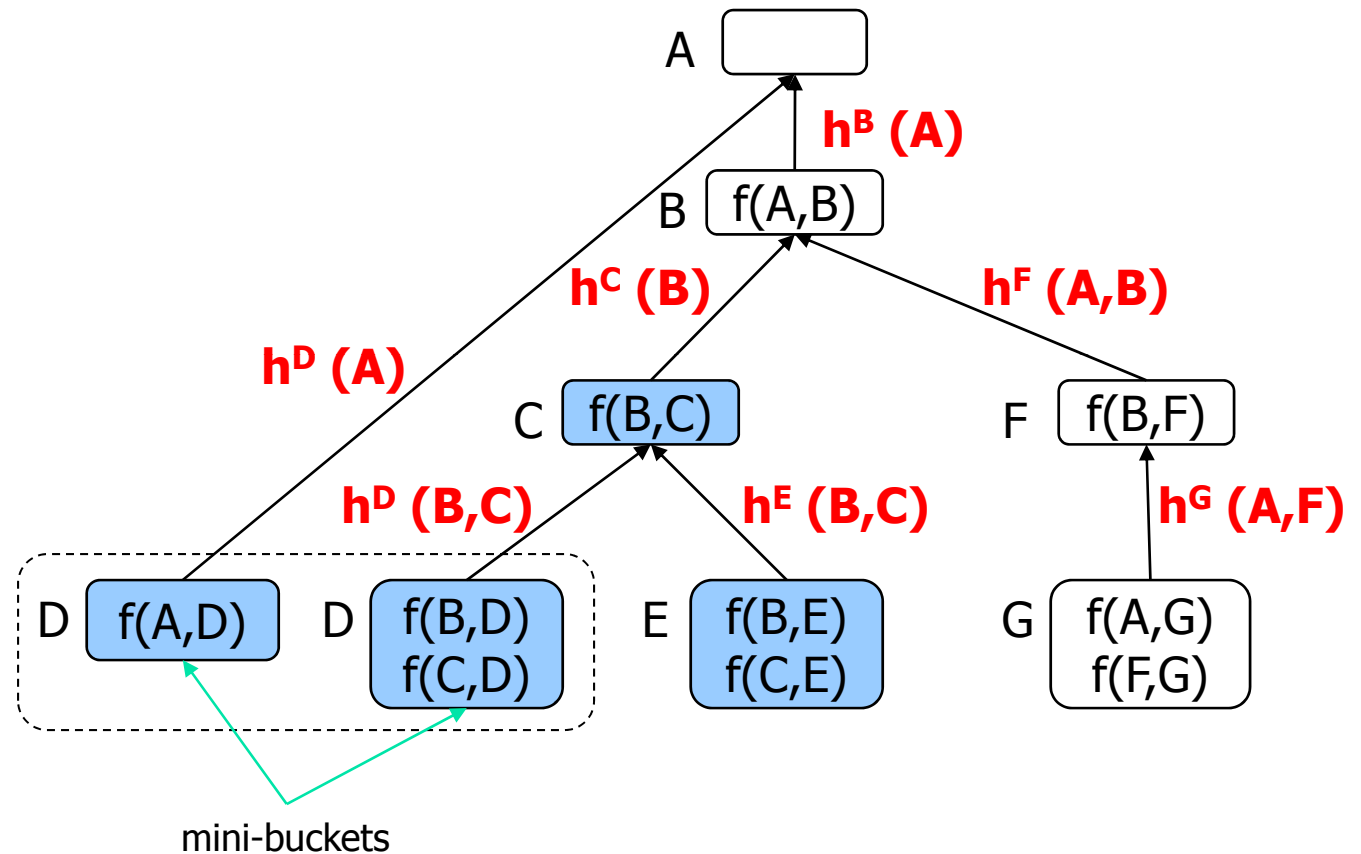


$$h^*(a, b, c) = h^D(a, b, c) + h^E(b, c)$$

Static Mini-Bucket Heuristics

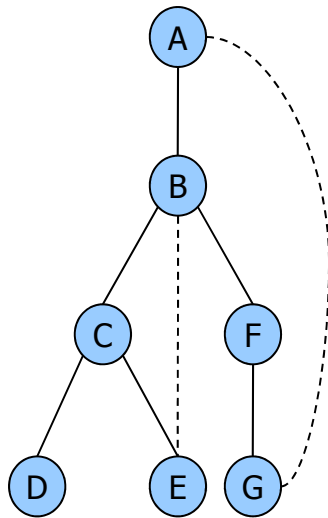
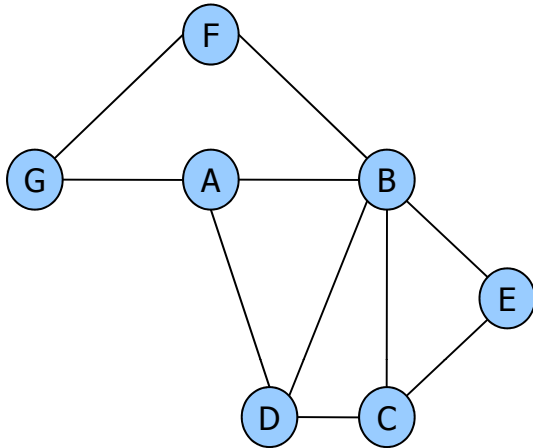


Ordering: (A, B, C, D, E, F, G)

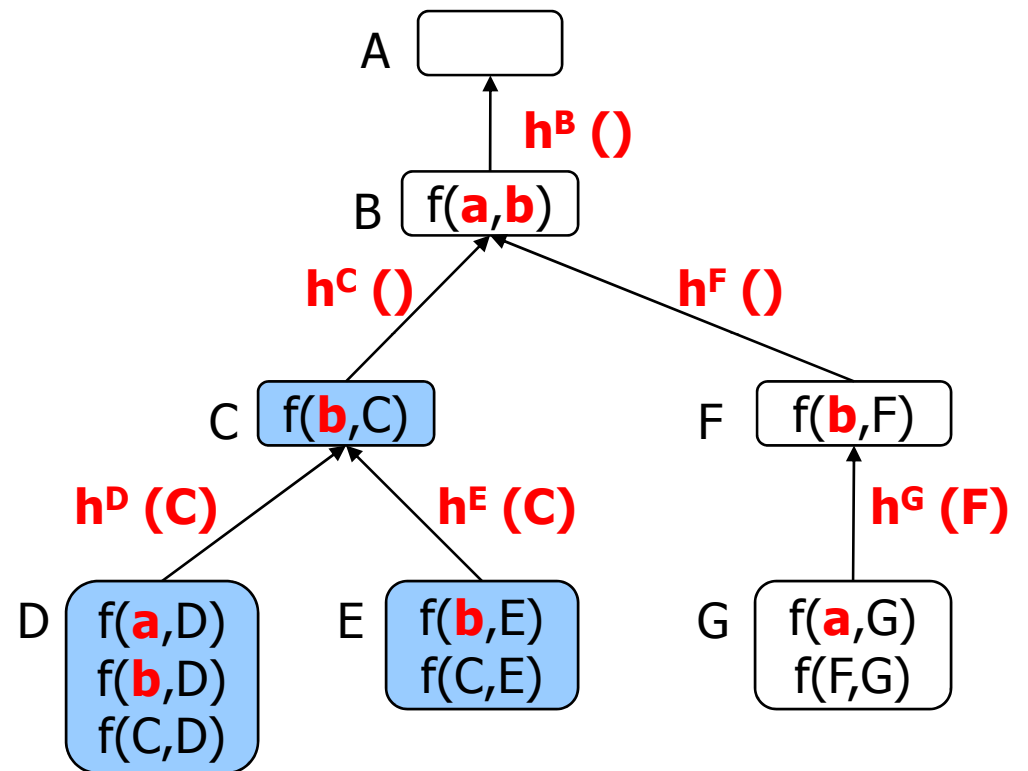


$$h(a, b, c) = h^D(a) + h^D(b, c) + h^E(b, c) \leq h^*(a, b, c)$$

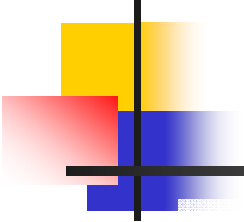
Dynamic Mini-Bucket Heuristics



Ordering: (A, B, C, D, E, F, G)

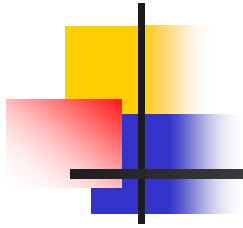


$$h(a, b, c) = h^D(c) + h^E(c) = h^*(a, b, c)$$



Outline

- Introduction
- Inference
- **Search**
 - **Exact**
 - AND/OR search trees
 - **AND/OR Branch-and-Bound search**
 - Lower bounding heuristics
 - Dynamic variable orderings
 - AND/OR search graphs (caching)
 - AND/OR search for 0-1 integer programming
 - AND/OR search for multi-objective optimization
 - Approximate: Sampling etc.
- Compilation: AND/OR Decision Diagrams
- Software



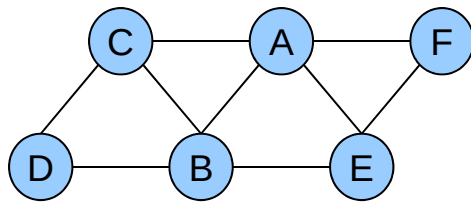
Dynamic Variable Orderings

(Marinescu & Dechter, ECAI'06)

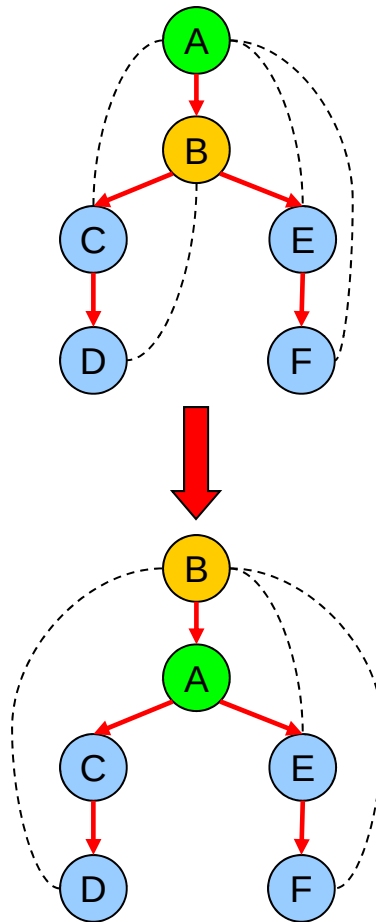
- Variable ordering heuristics:
 - **Semantic-based**
 - Aim at shrinking the size of the search space based on context and current value assignments
 - e.g. min-domain, min-dom/deg, min reduced cost
 - **Graph-based**
 - Aim at maximizing the problem decomposition
 - e.g. pseudo-tree arrangement

Orthogonal forces, use one as primary and break ties based on the other

Partial Variable Ordering



Primal graph



Variable Groups/Chains:

- {A,B}
- {C,D}
- {E,F}

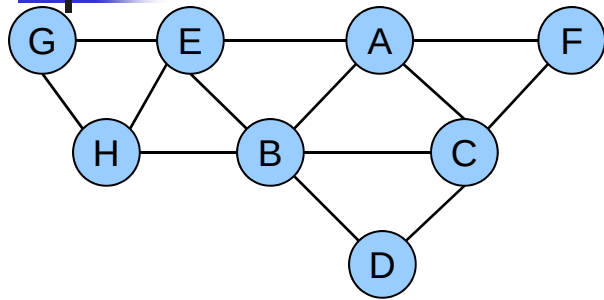
Instantiate {A,B}
before {C,D} and {E,F}

* {A,B} is a separator/chain

Variables on **chains**
in the pseudo tree
can be instantiated
dynamically, based
on some semantic
ordering heuristic

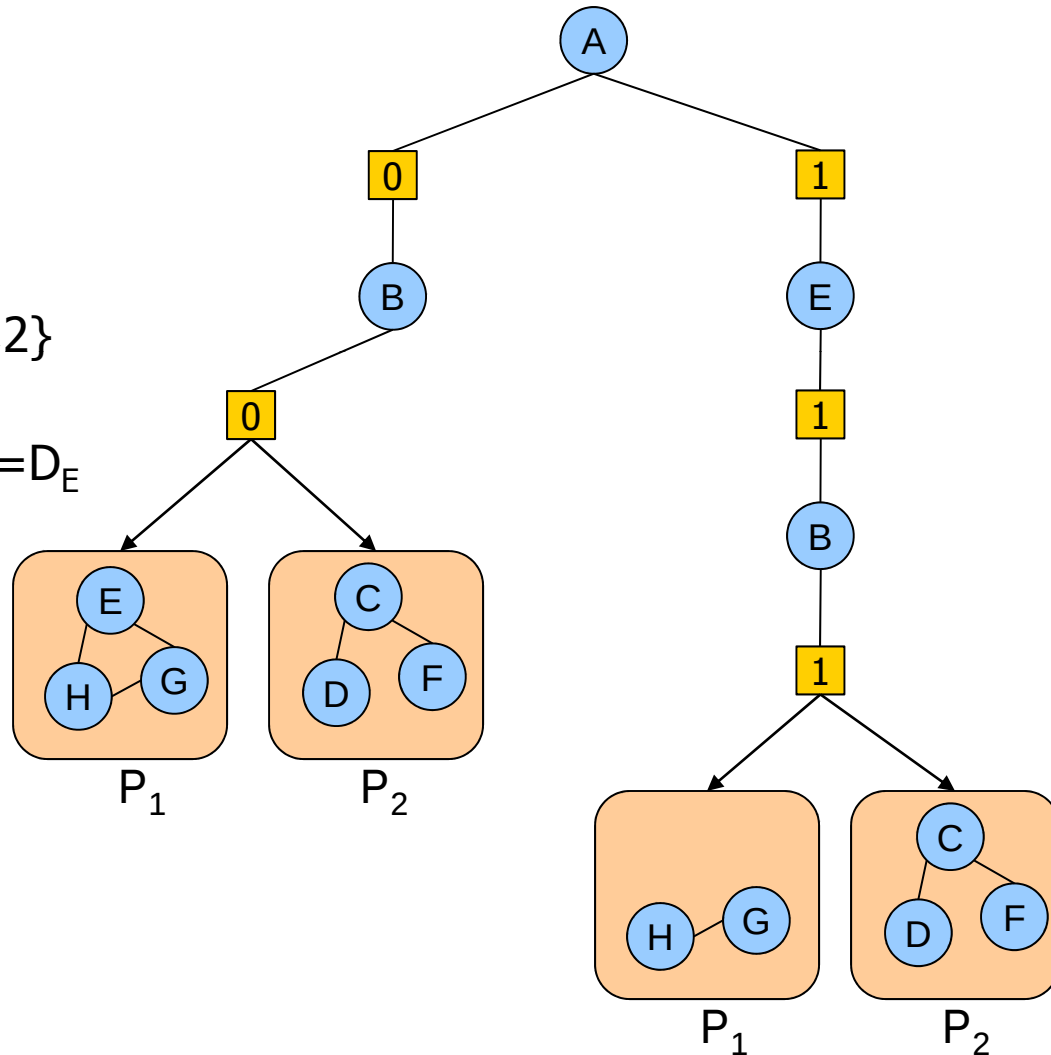
[similar idea is exploited by **BTD** (Jegou & Terrioux04)]

Full Dynamic Variable Ordering



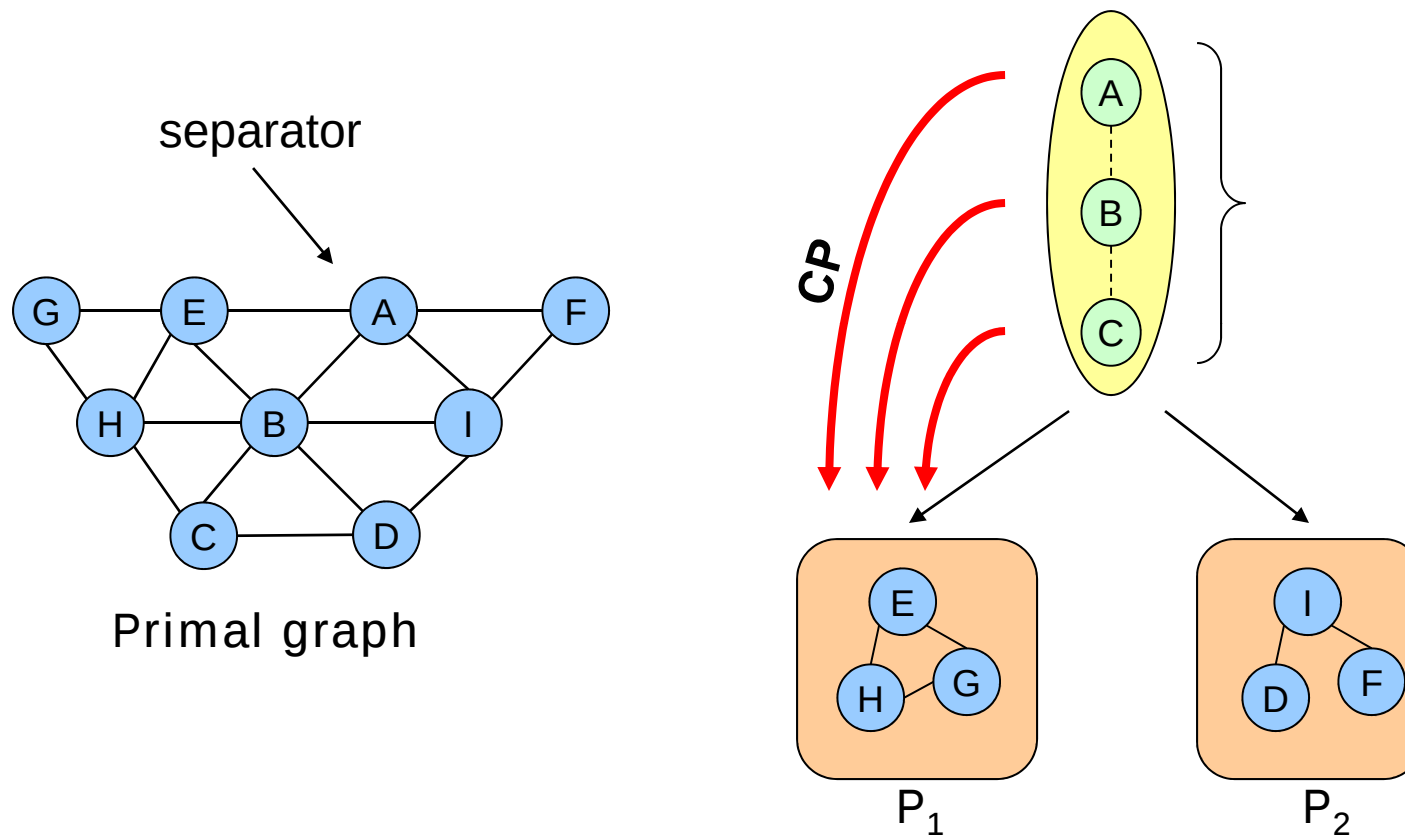
domains $D_A = \{0,1\}$ $D_B = \{0,1,2\}$
 $D_E = \{0,1,2,3\}$
 $D_C = D_D = D_F = D_G = D_H = D_I = \emptyset$

A	B	f(AB)	A	E	f(AE)
0	0	3	0	0	0
0	1	∞	0	1	5
0	2	∞	0	2	1
1	0	4	0	3	4
1	1	0	1	0	∞
1	2	6	1	1	∞
			1	2	0
			1	3	5

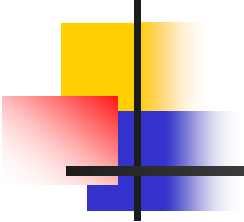


[similar idea exploited in #SAT (Bayardo & Pehoushek00)]

Dynamic Separator Ordering



Constraint Propagation may create **singleton** variables in **P1** and **P2** (changing the problem's structure), which in turn may yield smaller separators



Experiments

- Benchmarks
 - Belief Networks (BN)
 - Weighted CSPs (WCSP)
- Algorithms
 - **AOBB**
 - Samlam (BN)
 - Superlink (Genetic linkage analysis)
 - Toolbar (ie, DFBB+EDAC)
- Heuristics
 - Mini-Bucket heuristics (BN, WCSP)
 - EDAC heuristics (WCSP)

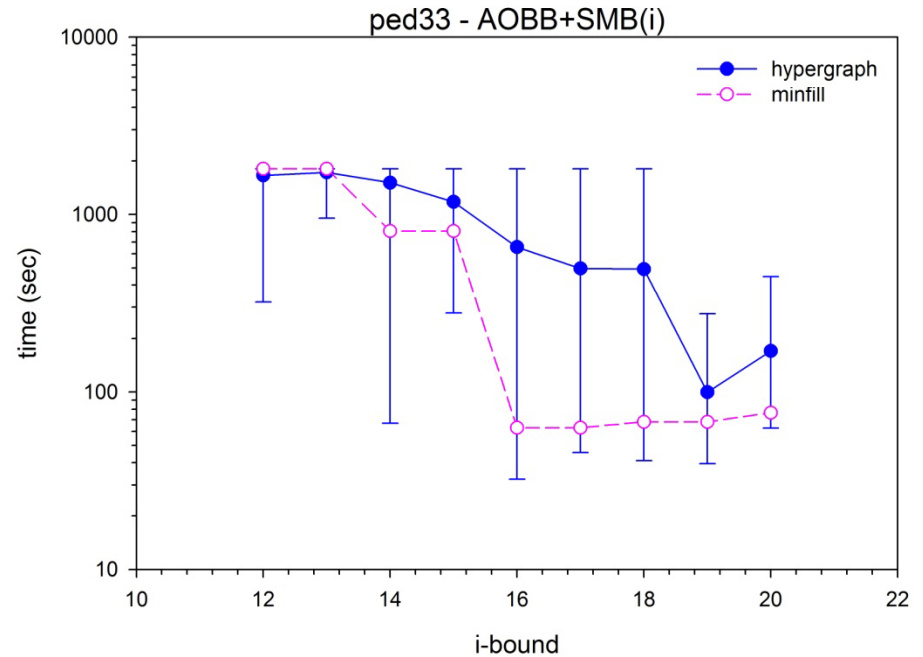
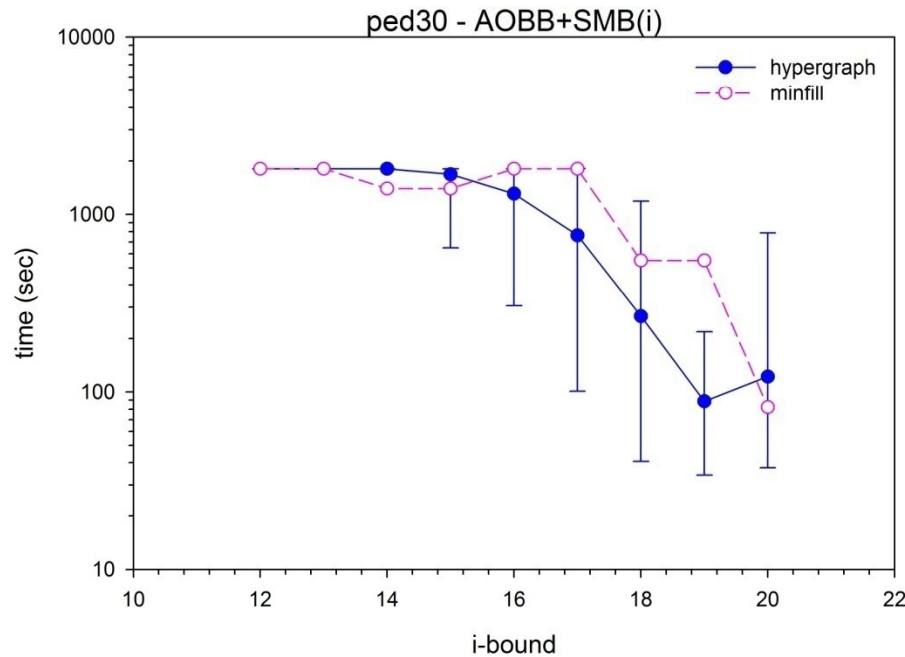
Genetic Linkage Analysis

(Fishelson&Geiger02)

pedigree (n, d) (w*, h)	Superlink v. 1.6 time	SamIam v. 2.3.2 time	MBE(i) BB+SMB(i) AOBB+SMB(i) i=12		MBE(i) BB+SMB(i) AOBB+SMB(i) i=16		MBE(i) BB+SMB(i) AOBB+SMB(i) i=20	
			time	nodes	time	nodes	time	nodes
ped18 (1184, 5) (21, 119)	139.06	157.05	0.51 - -	- - -	4.59 - 270.96	- - 2,555,078	19.30 - 20.27	- - 7,689
ped25 (994, 5) (29, 53)	-	out	0.34 - -	- - -	3.20 - -	- - -	33.42 - 1894.17	- - 11,709,153
ped30 (1016, 5) (25, 51)	13095.83	out	0.31 - 5563.22	- - 63,068,960	2.66 - 1811.34	- - 20,275,620	24.88 - 82.25	- - 588,558
ped33 (581, 5) (26, 48)	-	out	0.41 - 2335.28	- - 32,444,818	5.28 - 62.91	- - 807,071	51.24 - 76.47	- - 320,279
ped39 (1272, 5) (23, 94)	322.14	out	0.52 - -	- - -	8.41 - 4041.56	- - 52,804,044	81.27 - 141.23	- - 407,280

Min-fill pseudo tree. Time limit 3 hours.

Impact of the Pseudo Tree



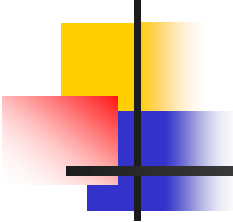
Runtime distribution for hypergraph pseudo trees over 20 independent runs.
ped30 and **ped33** linkage networks.

Dynamic Variable Orderings

(Bensana et al.99)

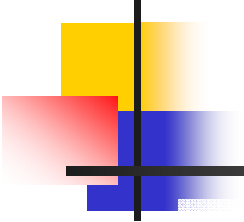
spot5	n	w*		toolbar	BBEDAC	AOEDAC	AOEDAC+PVO	DVO+AOEDAC	AOEDAC+DSO
	c	h							
29	16	7	time	4.56	109.66	613.79	545.43	0.83	11.36
	57	8	nodes	218,846	710,122	8,997,894	7,837,447	8,698	92,970
42b	14	9	time	-	-	-	-	-	6825.4
	75	9	nodes	-	-	-	-	-	27,698,614
54	14	9	time	0.31	0.97	31.34	9.11	0.06	0.75
	75	9	nodes	21,939	8,270	823,326	90,495	688	6,614
404	16	10	time	151.11	2232.89	255.83	152.81	12.09	1.74
	89	12	nodes	6,215,135	7,598,995	3,260,610	1,984,747	88,079	14,844
408b	18	10	time	-	-	-	-	-	747.71
	106	13	nodes	-	-	-	-	-	2,134,472
503	22	11	time	-	-	-	-	-	53.72
	131	15	nodes	-	-	-	-	-	231,480

SPOT5 benchmark. Time limit 2 hours.



Summary

- New generation of depth-first AND/OR Branch-and-Bound search
- Heuristics based on
 - Mini-Bucket approximation (static, dynamic)
 - Local consistency (FDAC, EDAC, VAC, ...)
- Dynamic variable orderings
- Superior to state-of-the-art solvers traversing the classic OR search space

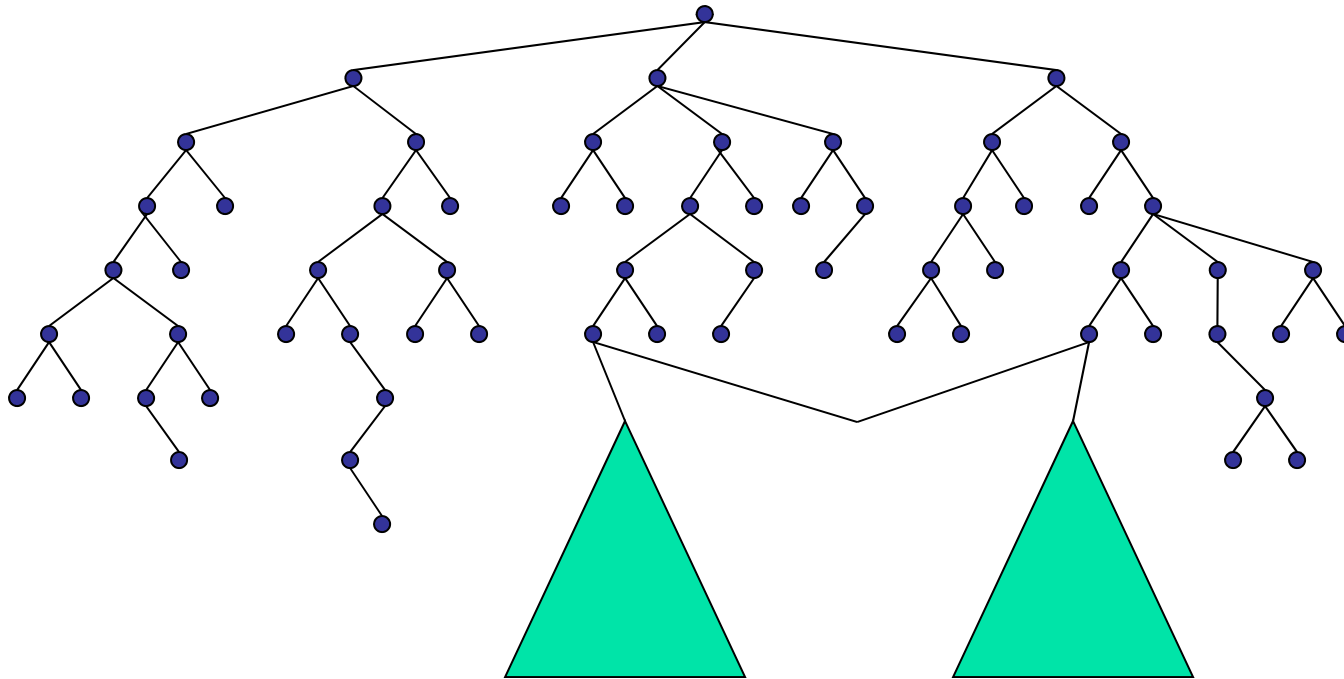


Outline

- Introduction
- Inference
- **Search**
 - **Exact**
 - AND/OR search trees
 - AND/OR Branch-and-Bound search
 - **AND/OR search graphs (caching)**
 - AND/OR Branch-and-Bound with caching
 - Best-First AND/OR search
 - AND/OR search for 0-1 integer programming
 - AND/OR search for multi-objective optimization
 - Approximate: Sampling etc.
- Compilation: AND/OR Decision Diagrams
- Software

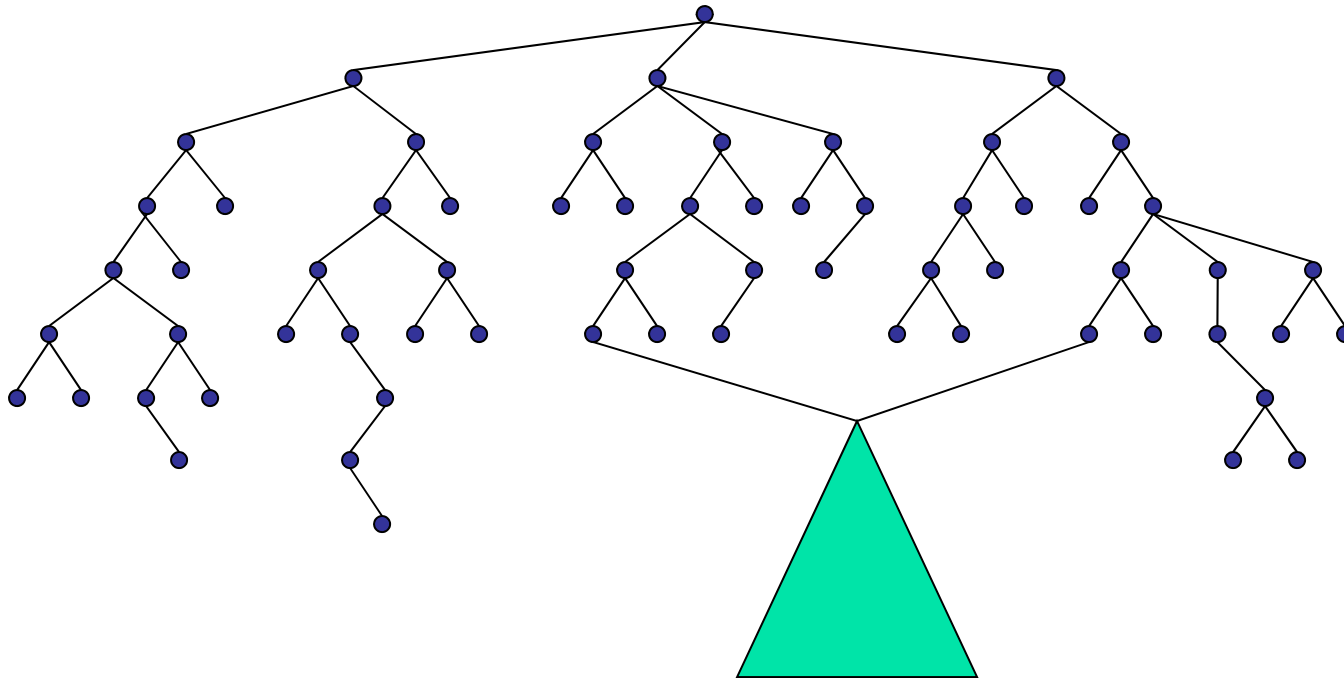
From search trees to search **graphs**

- Any two nodes that root identical subtrees (subgraphs) can be **merged**



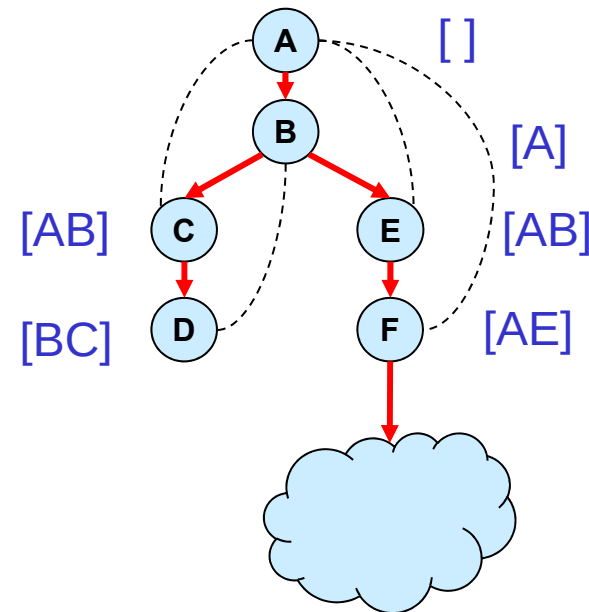
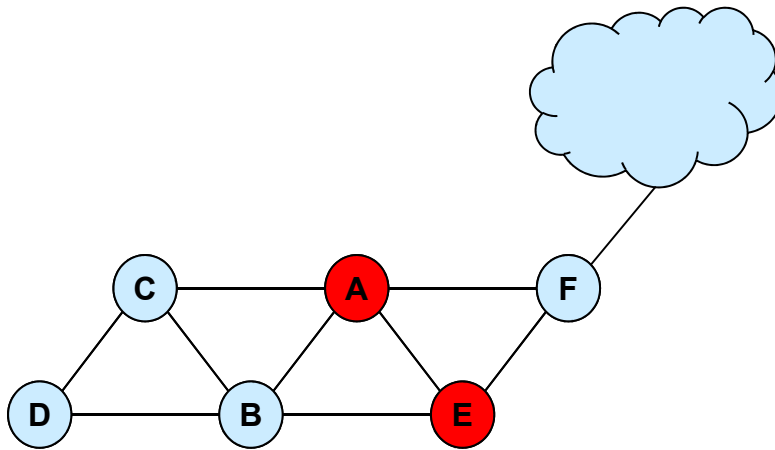
From search trees to search **graphs**

- Any two nodes that root identical subtrees (subgraphs) can be **merged**



Merging based on context

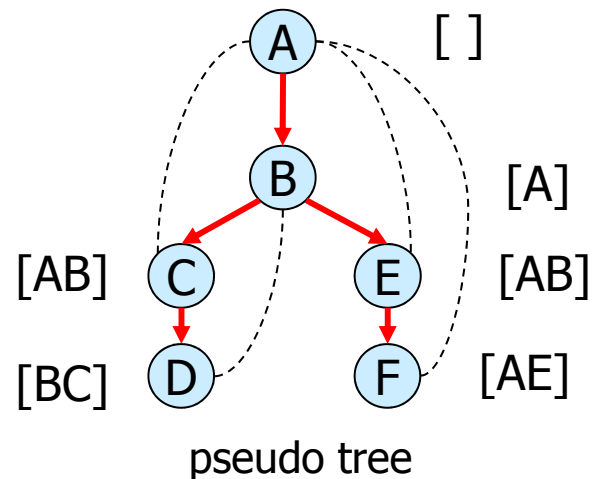
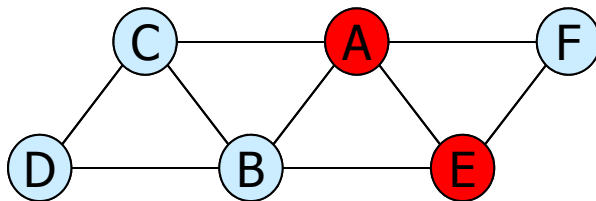
context (X) = ancestors of X connected to $\begin{cases} X \\ \text{descendants of } X \end{cases}$



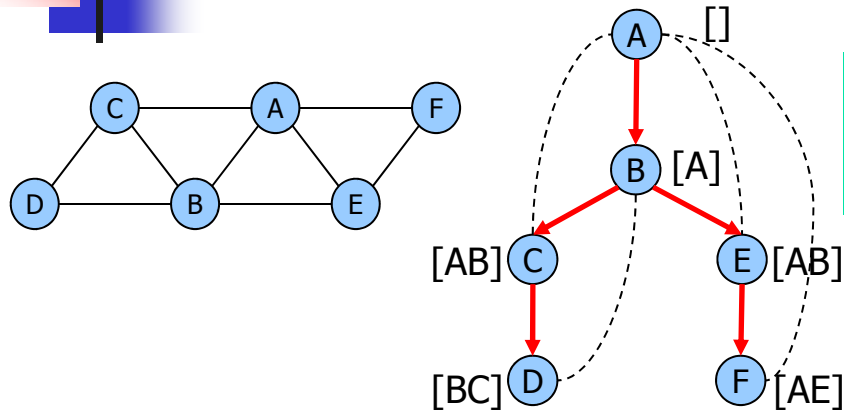
Merging Based on Context

- One way of recognizing nodes that can be merged (based on graph structure)

context(X) = ancestors of X in the pseudo tree that are connected to X, or to descendants of X

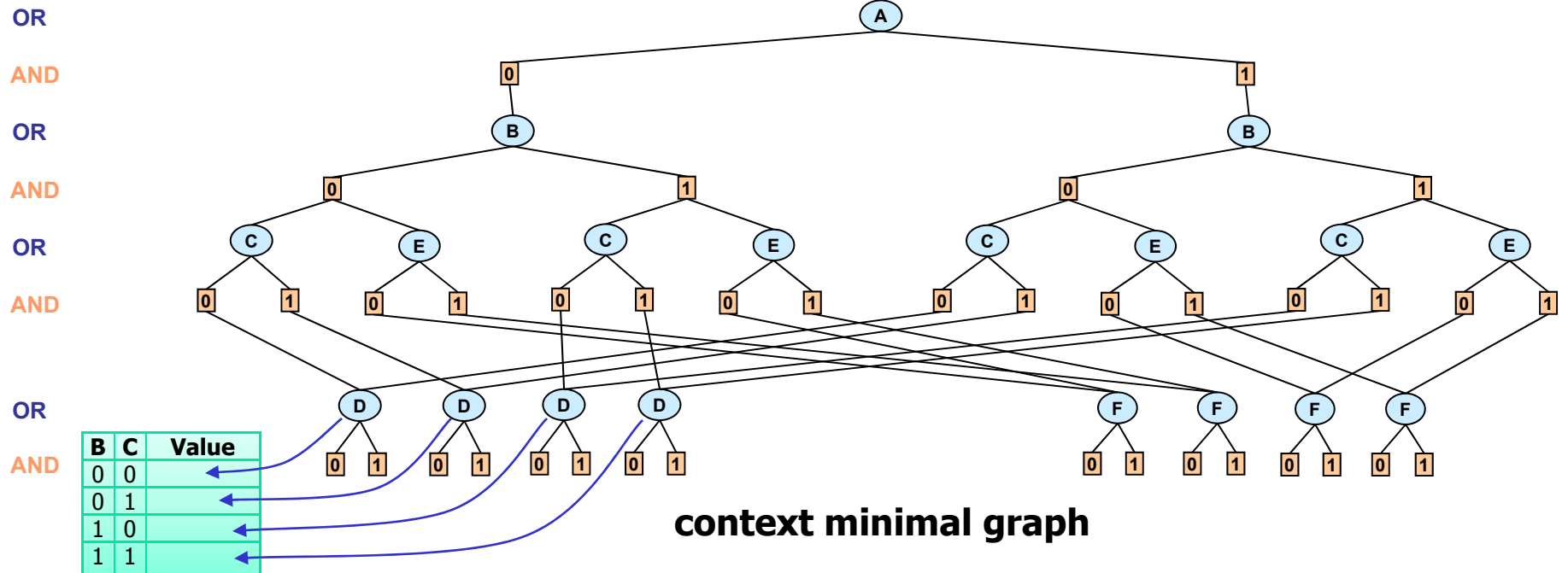


AND/OR Search Graph



A	B	f ₁	A	C	f ₂	A	E	f ₃	A	F	f ₄	B	C	f ₅	B	D	f ₆	B	E	f ₇	C	D	f ₈	E	F	f ₉
0	0	2	0	0	3	0	0	0	0	0	2	0	0	0	0	0	4	0	0	3	0	0	1	0	0	1
0	1	0	0	1	0	0	1	3	0	1	0	0	1	1	0	1	2	0	1	2	0	1	4	0	1	0
1	0	1	1	0	0	1	0	2	1	0	0	1	0	2	1	0	1	1	0	1	1	0	0	1	0	0
1	1	4	1	1	1	1	1	0	1	1	2	1	1	4	1	1	0	1	1	0	1	1	0	1	1	2

$$f(X) = \min_x \sum_{i=1}^9 f_i(X)$$



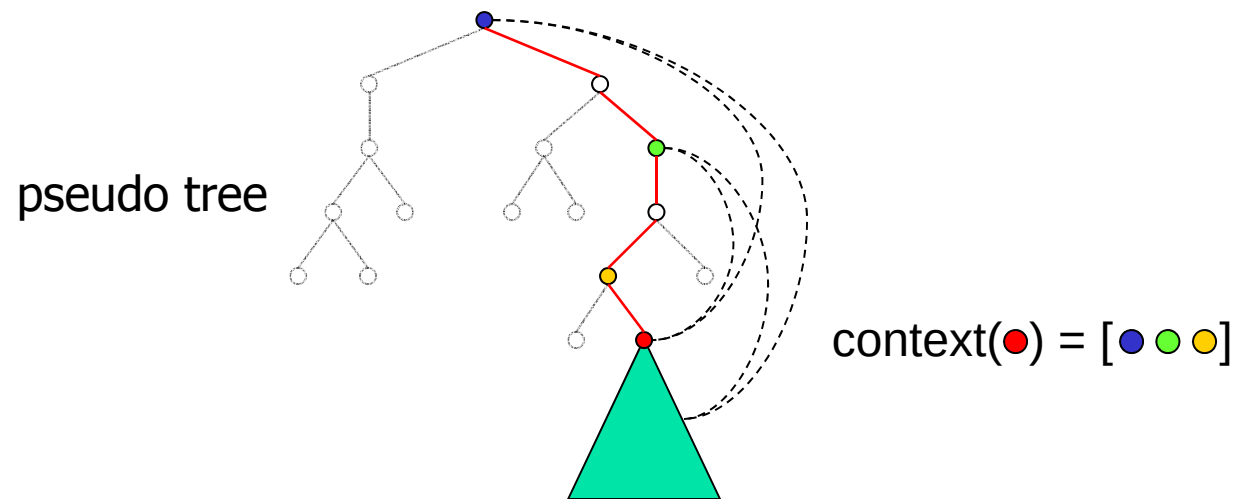
Cache table for D

How big is the context?

context (X) = ancestors of X in pseudo tree that are connected to X, or to descendants of X

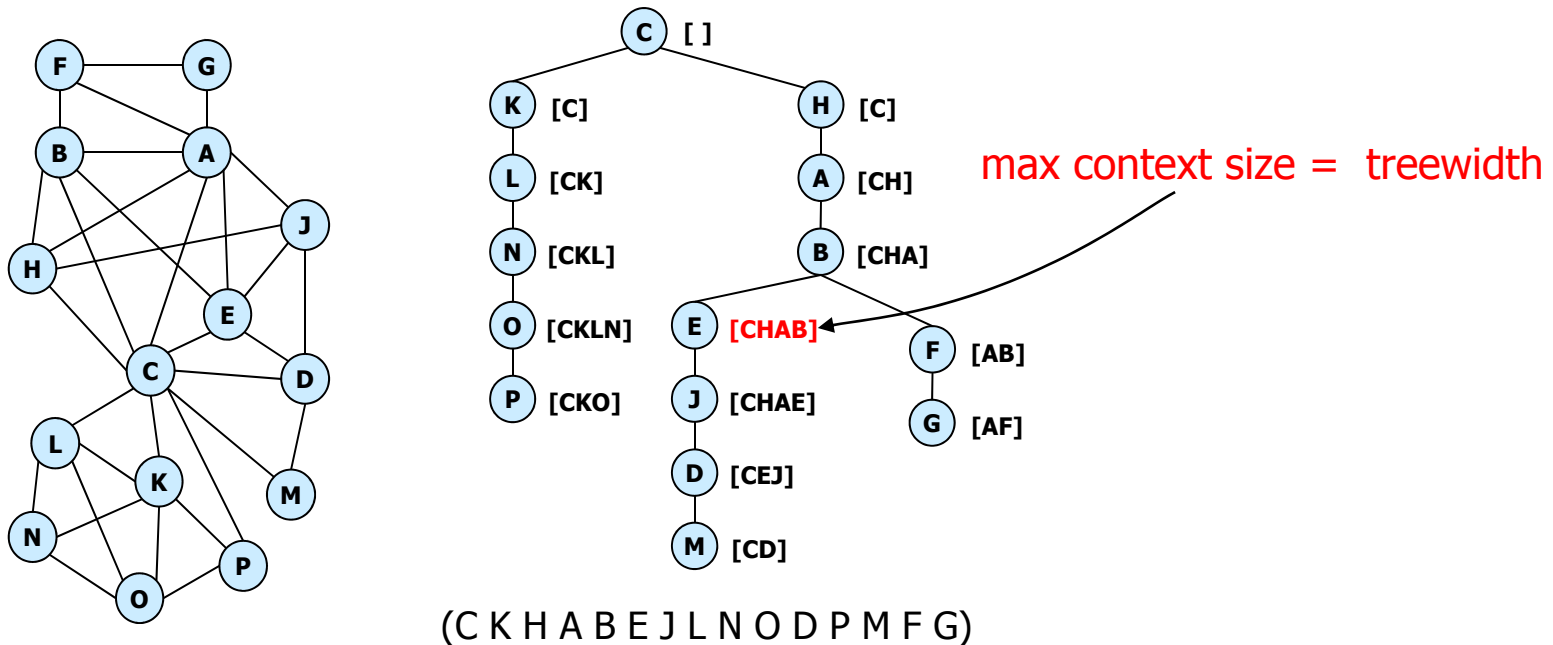
context (X) = parents in the induced graph

max |context| = induced width = treewidth

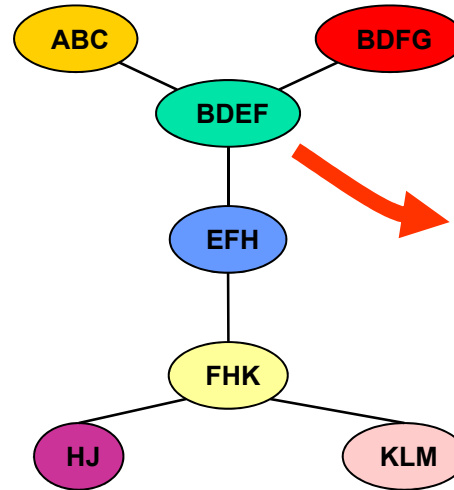
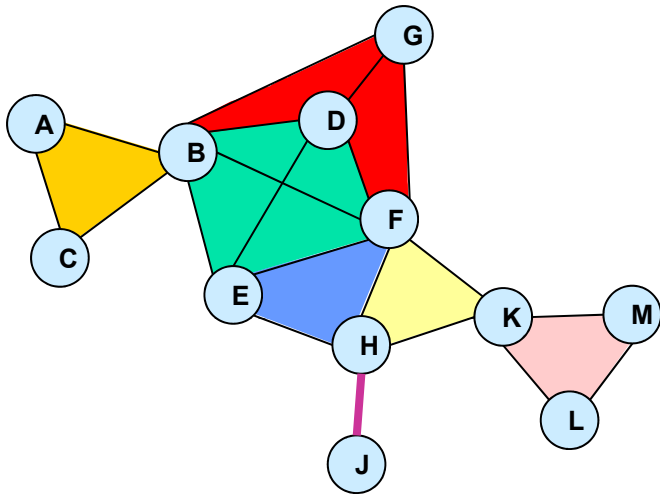


How Big Is The Context?

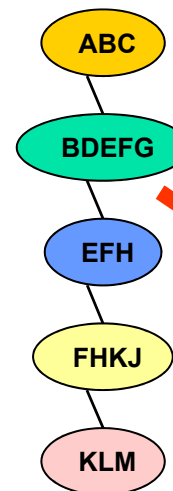
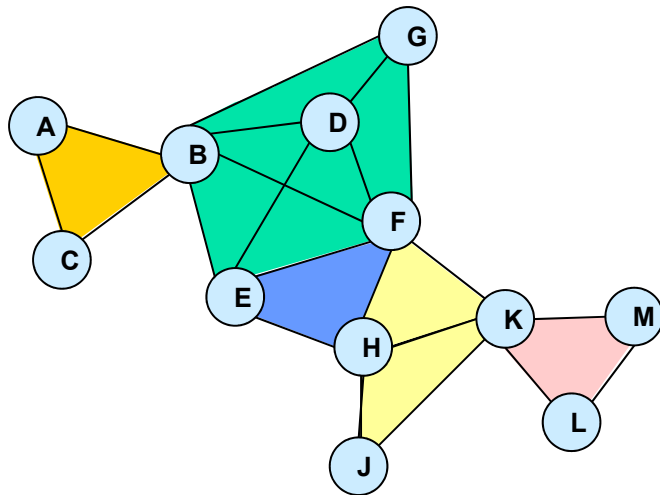
Theorem: The maximum **context** size for a pseudo tree **is equal** to the **treewidth** of the graph along the pseudo tree.



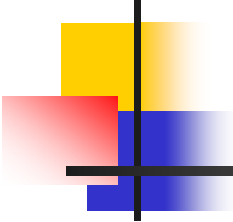
Treewidth vs. pathwidth



treewidth = 3
= (max cluster size) - 1



pathwidth = 4
= (max cluster size) - 1



Complexity of AND/OR Graph Search

	AND/OR graph	OR graph
Space	$O(n d^{w^*})$	$O(n d^{pw^*})$
Time	$O(n d^{w^*})$	$O(n d^{pw^*})$

d = domain size

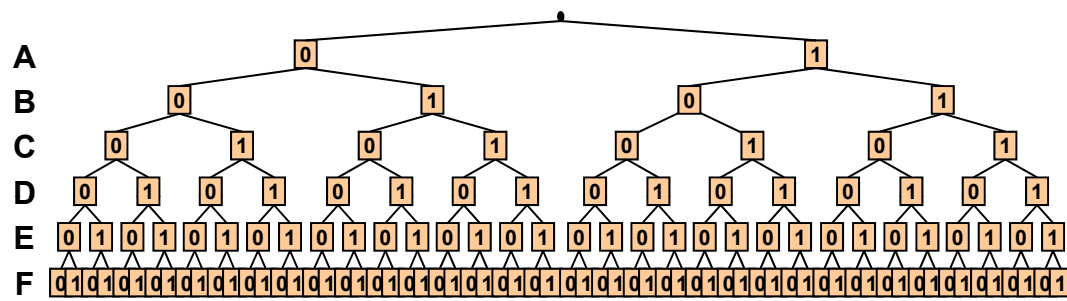
n = number of variables

w^* = treewidth

pw^* = pathwidth

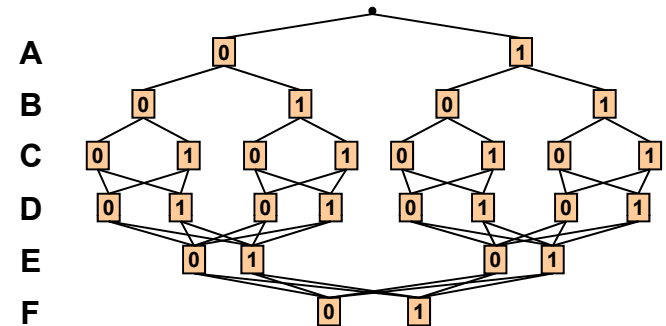
$$w^* \leq pw^* \leq w^* \log n$$

All Four Search Spaces



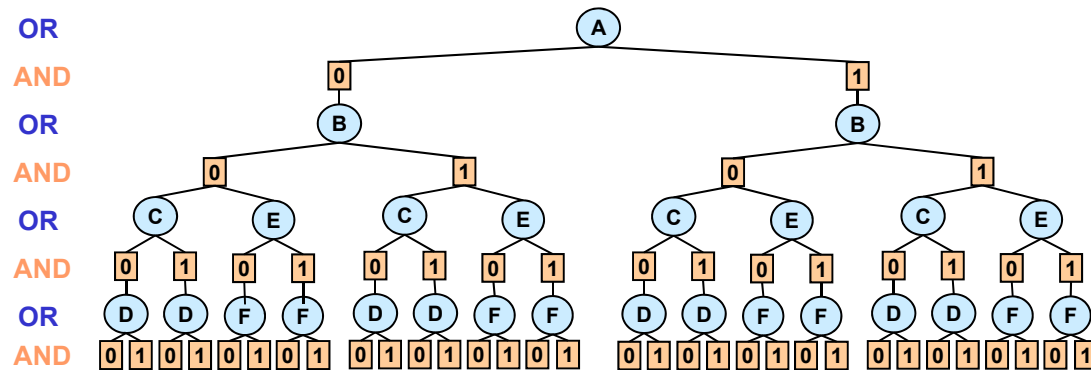
Full OR search tree

126 nodes



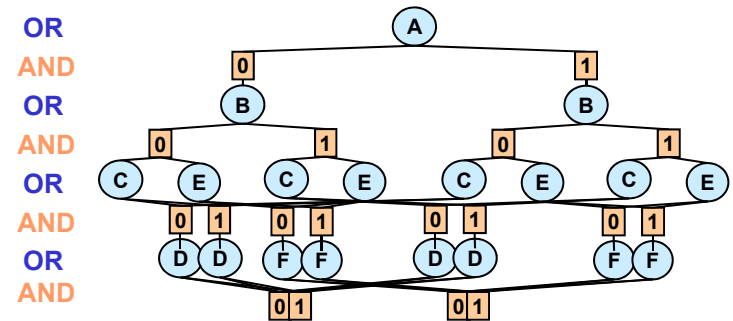
Context minimal OR search graph

28 nodes



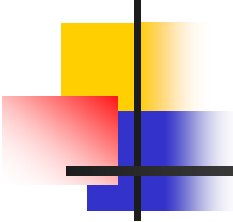
Full AND/OR search tree

54 AND nodes



Context minimal AND/OR search graph

18 AND nodes



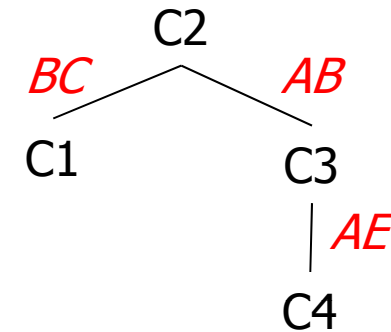
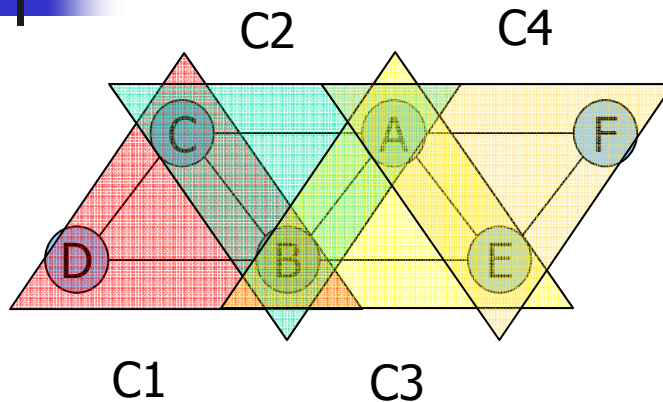
AND/OR Branch-and-Bound with Caching

(Marinescu & Dechter, AAAI'06)

- Associate each node n with a heuristic lower bound $h(n)$ on $v(n)$
- EXPAND (top-down)
 - Evaluate $f(T')$ and prune search if $f(T') \geq UB$
 - If not in cache, expand the tip node n
- PROPAGATE (bottom-up)
 - Update value of the parent p of n
 - OR nodes: minimization
 - AND nodes: summation
 - Cache value of n , based on context

Backtrack with Tree Decomposition

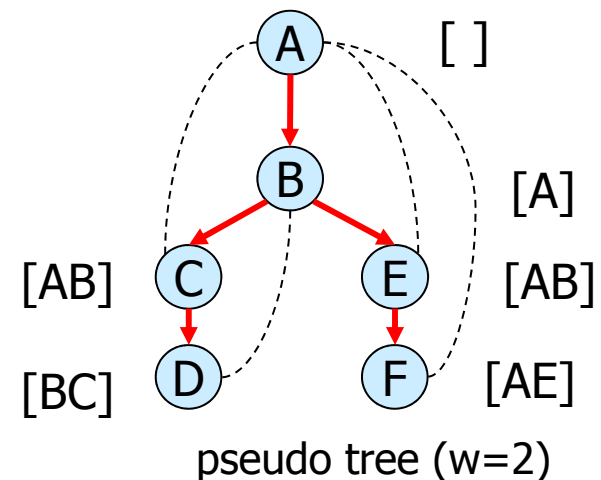
(Jegou & Terrioux, ECAI 2004)



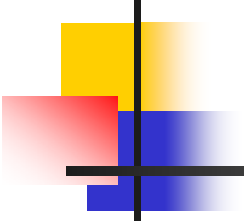
tree decomposition ($w=2$)

BTD:

- AND/OR graph search (caching on separators)
- Partial variable ordering (dynamic inside clusters)
- Maintaining local consistency



pseudo tree ($w=2$)



Backtrack with Tree Decomposition

- Before the search

- Merge clusters with a separator size $> p$
- Time $O(k \exp(w'))$, Space $O(\exp(p))$
- More freedom for variable ordering heuristics

- Properties

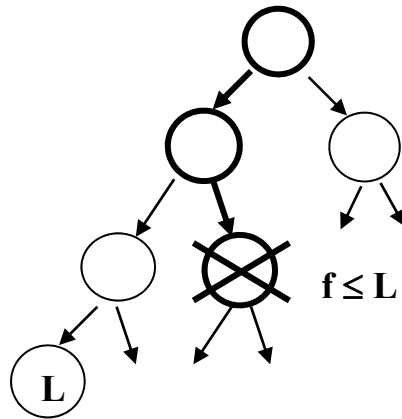
- $\text{BTD}(-1)$ is Depth-First Branch and Bound
- $\text{BTD}(0)$ solves connected components independently
- $\text{BTD}(1)$ exploits bi-connected components
- $\text{BTD}(s)$ is Backtrack with Tree Decomposition
s: largest separator size

Basic Heuristic Search Schemes

Heuristic function $f(x^p)$ computes a lower bound on the best extension of x^p and can be used to guide a heuristic search algorithm. We focus on:

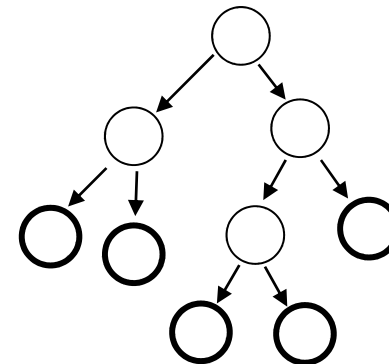
1. DF Branch-and-Bound

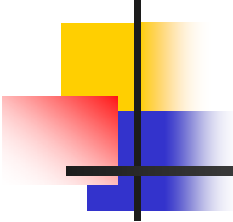
Use heuristic function $f(x^p)$ to prune the depth-first search tree
Linear space



2. Best-First Search

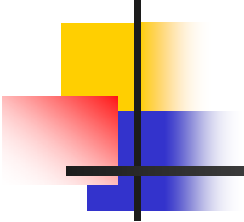
Always expand the node with the highest heuristic value $f(x^p)$
Needs lots of memory





Best-First Principle

- Best-first search expands first the node with the best heuristic evaluation function among all node encountered so far
- It **never** expands nodes whose cost is beyond the optimal one, unlike depth-first search algorithms (Dechter & Pearl, 1985)
- Superior among memory intensive algorithms employing the **same heuristic function**

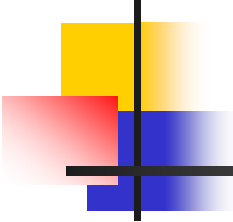


Best-First AND/OR Search (AOBF)

(Marinescu & Dechter, CPAIOR'07, AAAI'07, UAI'07)

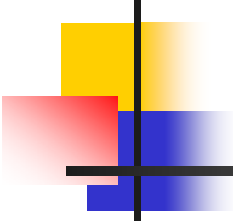
- **Maintains the set of best partial solution trees**
- **Top-down Step (EXPAND)**
 - Traces down marked connectors from root
 - i.e., **best partial solution tree**
 - Expands a tip node **n** by generating its successors **n'**
 - Associate each successor with heuristic estimate **h(n')**
 - Initialize **v(n') = h(n')**
- **Bottom-up Step (REVISE)**
 - Updates node values **v(n)**
 - OR nodes: **minimization**
 - AND nodes: **summation**
 - Marks the most promising solution tree from the root
 - Label the nodes as SOLVED:
 - OR is SOLVED if marked child is SOLVED
 - AND is SOLVED if all children are SOLVED
- **Terminate when root node is SOLVED**

(specializes Nilsson's AO* to solving COP) (Nilsson, 1984)



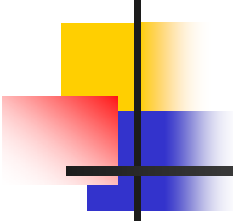
AOBF versus AOBB

- **AOBF** with the same heuristic as **AOBB** is likely to expand the smallest search space
- **AOBB** improves its heuristic function dynamically, whereas **AOBF** uses only $h(n)$
- **AOBB** can use far less memory by avoiding for example dead-caches, whereas **AOBF** keeps in memory the explicated search graph
- **AOBB** is (almost) any-time, whereas **AOBF** is not



Lower Bounding Heuristics

- **AOBF** can be guided by:
 - Static Mini-Bucket heuristics
(Kask & Dechter, AIJ'01), (Marinescu & Dechter, IJCAI'05)
 - Dynamic Mini-Bucket heuristics
(Marinescu & Dechter, IJCAI'05)
 - LP Relaxations
(Nemhauser & Wolsey, 1988)



Experiments

- Benchmarks
 - Belief Networks (BN)
 - Weighted CSPs (WCSP)
- Algorithms
 - **AOBB-C** – AND/OR Branch-and-Bound w/ caching
 - **AOBF-C** – Best-first AND/OR Search
 - Samlam
 - Superlink
 - Toolbar (DFBB+EDAC), Toolbar-BTD (BTD+EDAC)
- Heuristics
 - Mini-Bucket heuristics

Genetic Linkage Analysis

(Fishelson & Geiger02)

pedigree (w*, h) (n, d)	SamIam Superlink	MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=12		MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=14		MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=16		MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=18	
		time	nodes	time	nodes	time	nodes	time	nodes
		0.42		0.83		1.78		5.75	
ped30 (23, 118) (1016, 5)	out 13095.83	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	214.10	1,379,131
		10212.70	93,233,570	8858.22	82,552,957	-	-	34.19	193,436
ped33 (37, 165) (581, 5)	out -	0.58		2.31		7.84		33.44	
		-	-	-	-	-	-	-	-
		2804.61	34,229,495	737.96	9,114,411	3896.98	50,072,988	159.50	1,647,488
ped42 (25, 76) (448, 5)	out 561.31	1426.99	11,349,475	307.39	2,504,020	1823.43	14,925,943	86.17	453,987
		out		140.61	407,387	out		74.86	134,068
		4.20		31.33		96.28		out	
ped42 (25, 76) (448, 5)	out 561.31	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	-	-
		-	-	-	-	2364.67	22,595,247		
ped42 (25, 76) (448, 5)	out 561.31	out		out		133.19	93,831		

Min-fill pseudo tree. Time limit 3 hours.

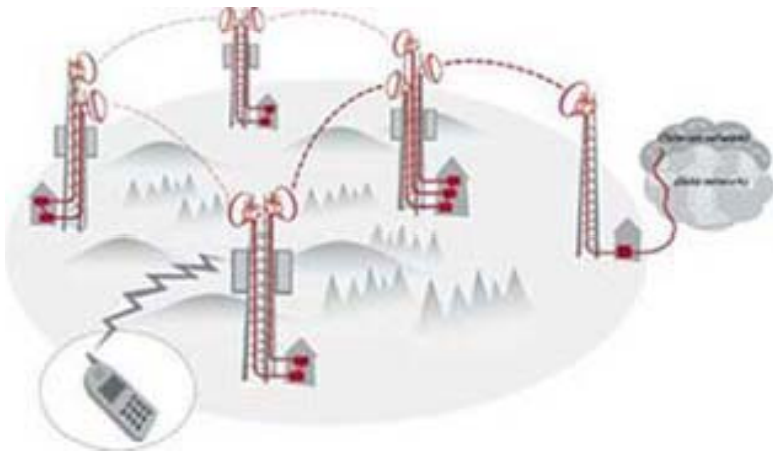
Mastermind Games

mastermind (w*, h) (n, r, k)	MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=12		MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=14		MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=16		MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=18	
	time	nodes	time	nodes	time	nodes	time	nodes
mm-04-08-04 (39, 103) (2616, 3, 2)	1.36 - 494.50 114.02 38.55	- - 744,993 82,070 33,069	2.08 - 270.60 66.84 29.19	- - 447,464 61,328 26,729	4.86 - 506.74 93.50 44.95	- - 798,507 79,555 38,989	16.53 - 80.86 30.80 20.64	- - 107,463 13,924 3,957
mm-03-08-05 (41, 111) (3692, 3, 2)	2.34 - - - out	- - - - -	8.52 - - - out	- - - - -	8.31 - - - 473.07	- - - - 199,725	24.94 - 1084.48 117.39 36.99	- - 1,122,008 55,033 8,297
mm-10-08-03 (51, 132) (2606, 3, 2)	1.64 - 161.35 19.86 4.80	- - 290,594 14,518 3,705	3.09 - 99.09 19.47 8.16	- - 326,662 14,739 4,501	7.55 - 89.06 22.34 11.17	- - 151,128 13,557 3,622	21.08 - 84.16 29.80 24.67	- - 127,130 9,388 3,619

Min-fill pseudo trees. Time limit 1 hour.
 toolbar, toolbar-BTD were not able to solve any instance.

Radio Link Frequency Assignment Problem

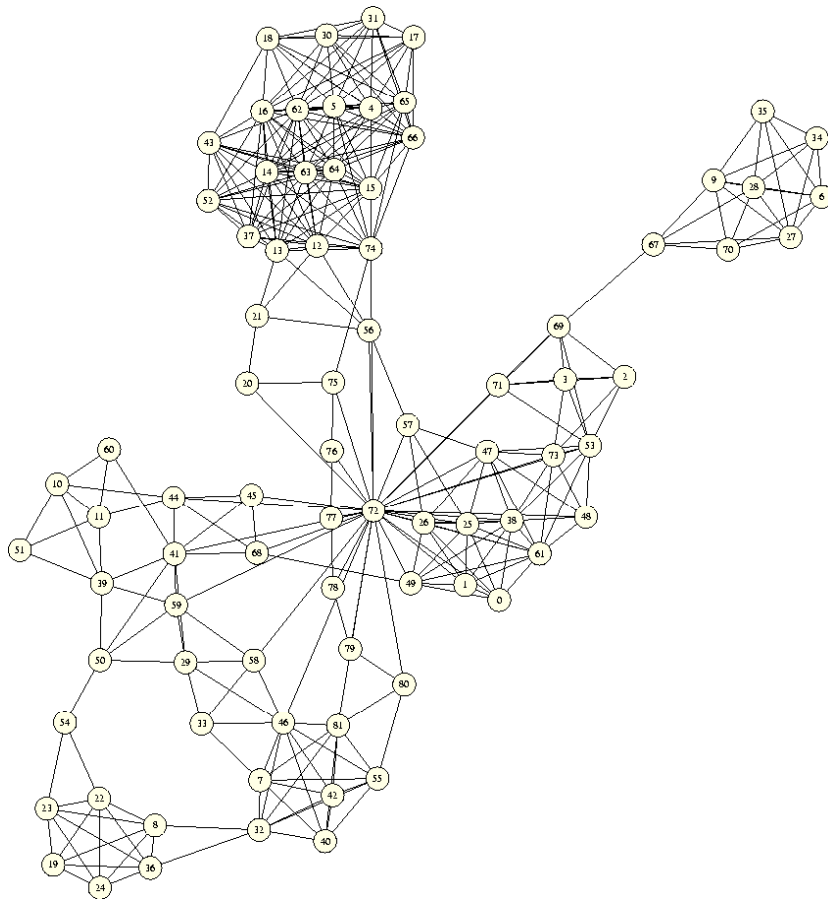
(Cabon et al., *Constraints* 1999) (Koster et al., *4OR* 2003)



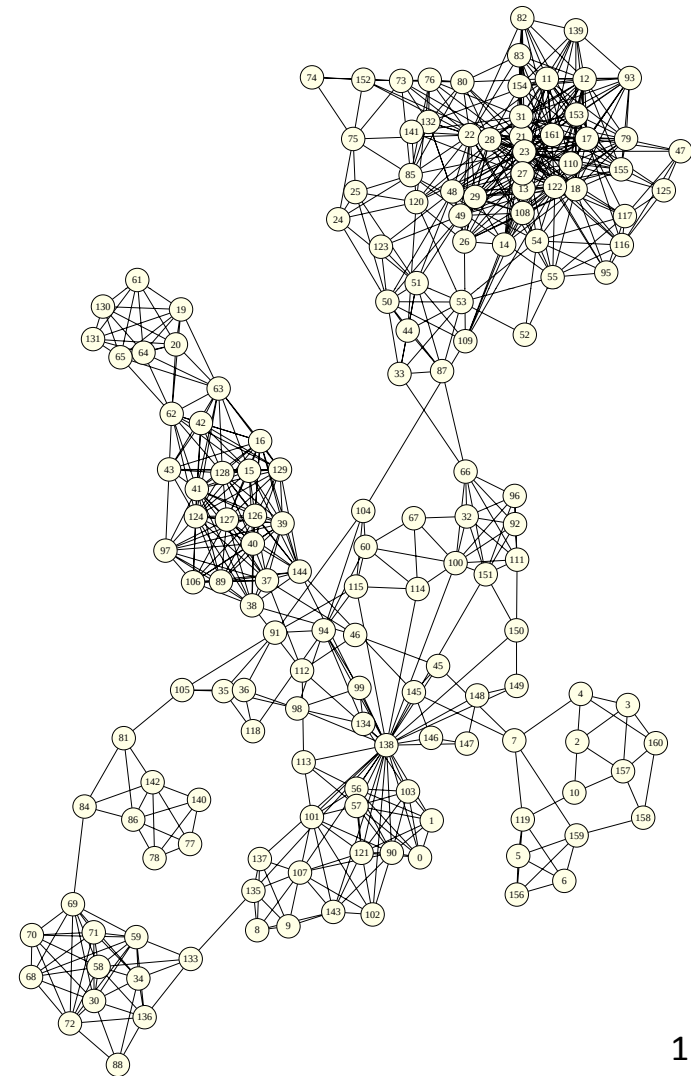
- Given a telecommunication network
 - ...find the **best** frequency for each communication link, avoiding interferences
-
- **Best** can be:
 - Minimize the maximum frequency, no interference (max)
 - **Minimize the global interference (sum)**
 - Generalizes graph coloring problems: $|f_1 - f_2| \geq a$

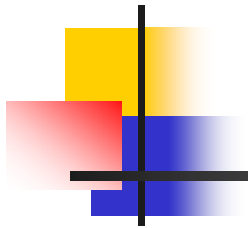
CELAR problem size: $n=100—458$; $d=44$; $m=1,000—5,000$

■ CELAR SCEN-06
n=100, d=44,
m=350, optimum=3389



■ CELAR SCEN-07r
n=162, d=44,
m=764, optimum=343592





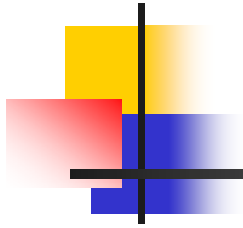
CELAR

toulbar2 v0.8 running on a 2.6 GHz computer with 32 GB

(Sanchez et al., IJCAI 2009)

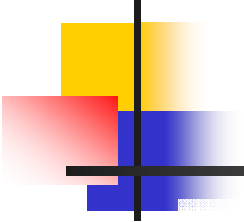
- Maximum Cardinality Search tree decomposition heuristic
- Root selection: largest (SCEN-06) / most costly (SCEN-07) cluster
- Last-conflict variable ordering and dichotomic branching
- Closed 1 open problem by exploiting tree decomposition and EDAC

CELAR	n	d	m	k	p	w	DFBB	BTD	RDS-BTD
SCEN-06	100	44	350	∞	∞	11	2588 sec.	221 sec.	316 sec.
SCEN-07r	162	44	764	354008	3	53	- > 50days	6 days	4.5 days



Summary

- New memory intensive AND/OR search algorithms for optimization in graphical models
- Depth-first and best-first control strategies
- Superior to state-of-the-art OR and AND/OR Branch-and-Bound tree search algorithms



Outline

- Introduction
- Inference
- **Search**
 - Exact
 - AND/OR search trees
 - AND/OR Branch-and-Bound search
 - AND/OR search graphs (caching)
 - **AND/OR search for 0-1 integer programming**
 - AND/OR search for multi-objective optimization
 - Approximate: Sampling etc.
- Compilation: AND/OR Decision Diagrams
- Software

0-1 Integer Linear Programming

minimize : $z = c_1x_1 + c_2x_2 + \dots + c_nx_n$

subject to :

$$a_1^1x_1 + a_2^1x_2 + \dots + a_n^1x_n \leq b^1$$

$$a_1^2x_1 + a_2^2x_2 + \dots + a_n^2x_n \leq b^2$$

...

$$a_1^mx_1 + a_2^mx_2 + \dots + a_n^mx_n \leq b^m$$

$$x_1, x_2, \dots, x_n \in \{0,1\}$$

:

- VLSI circuit design
- Scheduling
- Routing
- Combinatorial auctions
- Facility location
- ...

minimize : $z = 7A + 3B - 2C + 5D - 6E + 8F$

subject to :

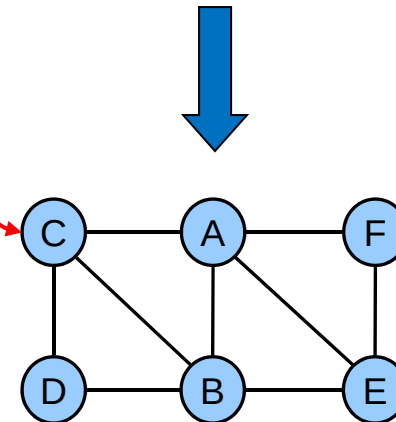
$$3A - 12B + C \leq 3$$

$$-2B + 5C - 3D \leq -2$$

$$2A + B - 4E \leq 2$$

$$A - 3E + F \leq 1$$

$$A, B, C, D, E, F \in \{0,1\}$$



primal graph

subject to :

$$-2B + 5C - 3D \leq -2$$

$$A - 3E + F \leq 1$$

$$A, B, C, D, E, F \in \{0, 1\}$$



AND

OR

AND

OR

AND

OR

AND

Weighted AND/OR Search Tree

minimize : $z = 7A + 3B - 2C + 5D - 6E + 8F$

subject to :

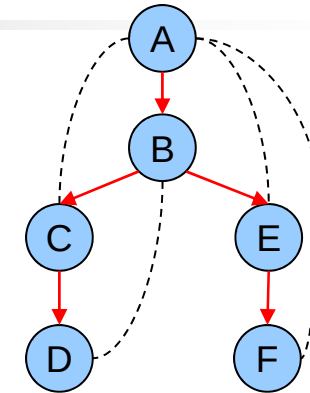
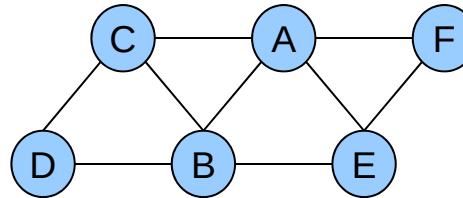
$$3A - 12B + C \leq 3$$

$$-2B + 5C - 3D \leq -2$$

$$2A + B - 4E \leq 2$$

$$A - 3E + F \leq 1$$

$$A, B, C, D, E, F \in \{0,1\}$$



OR

AND

OR

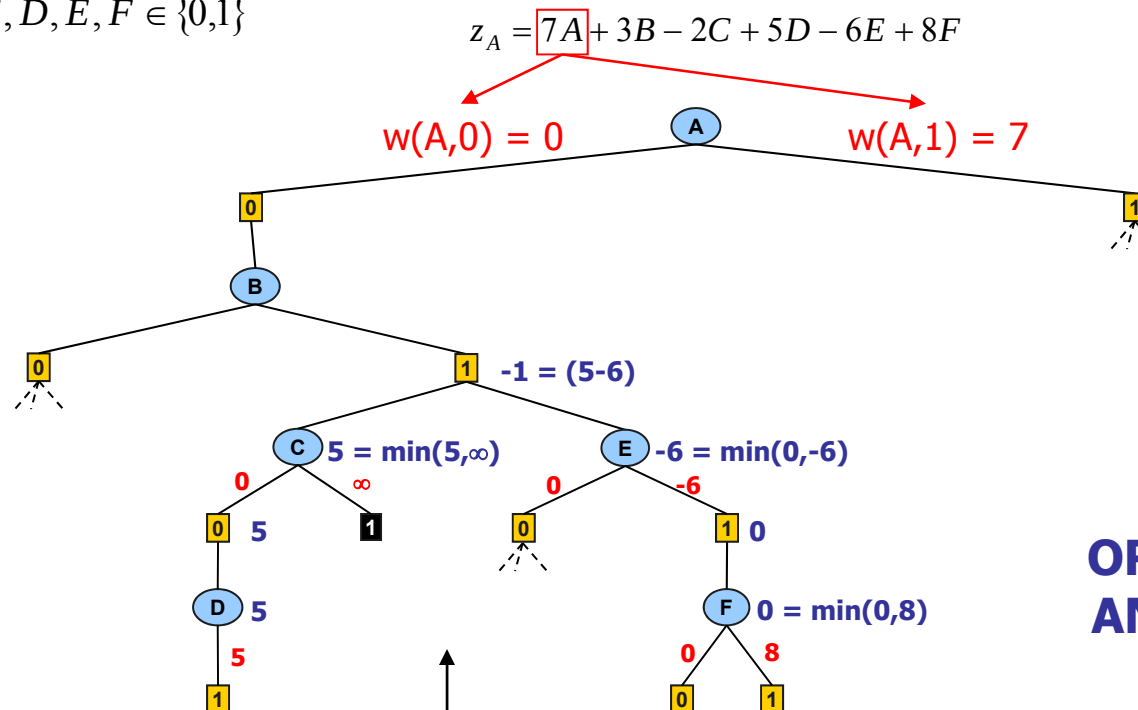
AND

OR

AND

OR

AND



**Node Value
(bottom up)**

**OR – minimization
AND – summation**

AND/OR Search Graph

minimize : $z = 7A + 3B - 2C + 5D - 6E + 8F$

subject to :

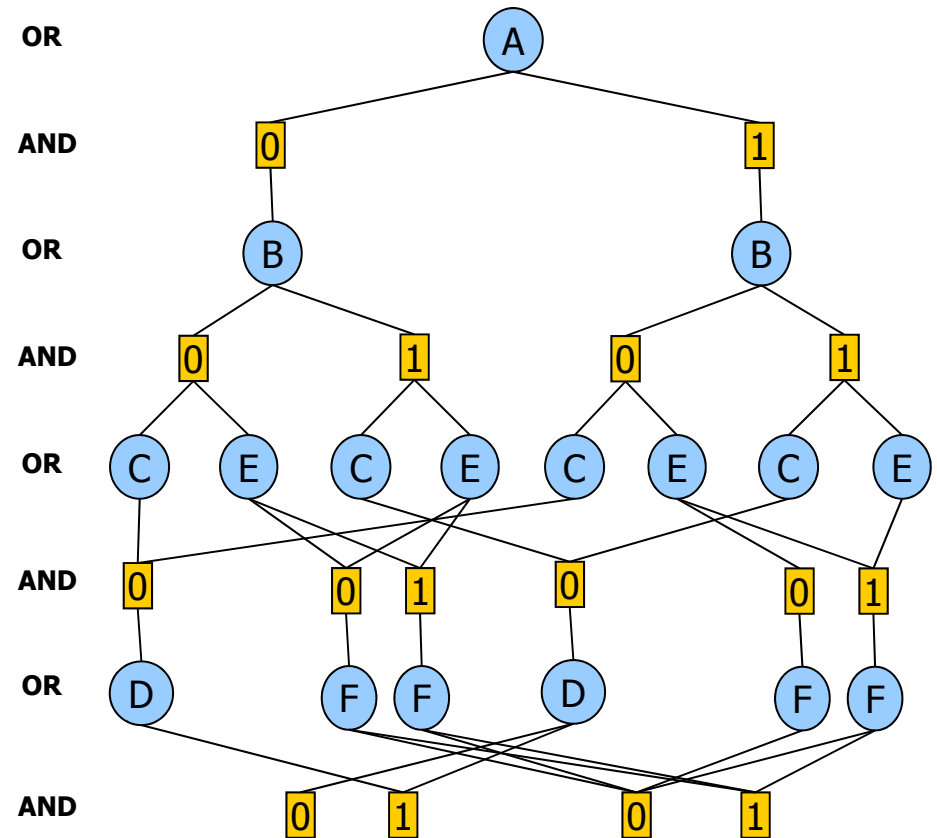
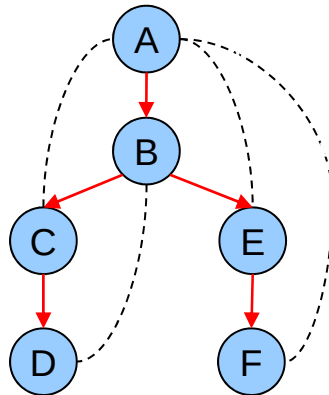
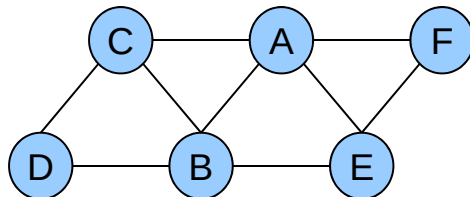
$$3A - 12B + C \leq 3$$

$$-2B + 5C - 3D \leq -2$$

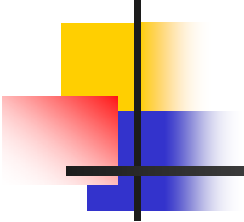
$$2A + B - 4E \leq 2$$

$$A - 3E + F \leq 1$$

$$A, B, C, D, E, F \in \{0, 1\}$$



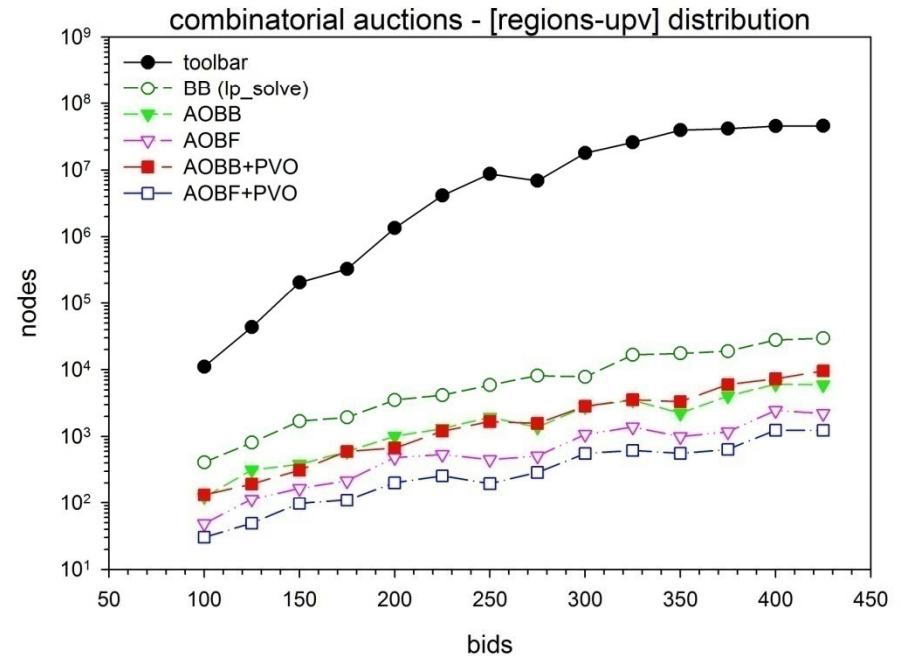
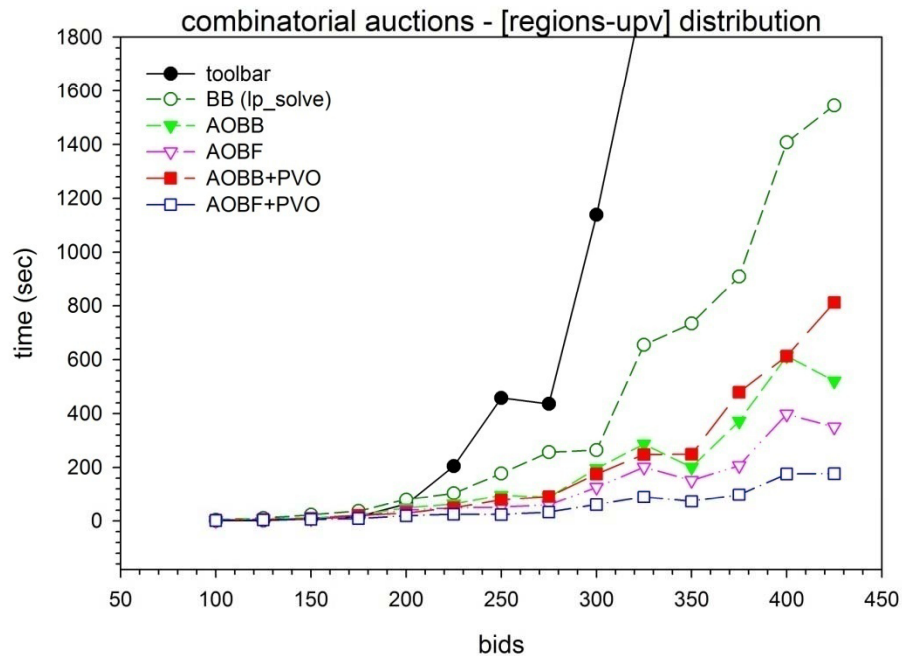
16 nodes (graph) vs. 54 nodes (tree)



Experiments

- Algorithms
 - AOBB, AOBF – tree search
 - AOBB+PVO, AOBF+PVO – tree search
 - AOBB-C, AOBF-C – graph search
 - Ip_solve 5.5, CPLEX 11.0, toolbar (DFBB+EDAC)
- Benchmarks
 - Combinatorial auctions
 - MAX-SAT instances
- Implementation
 - LP relaxation solved by Ip_solve 5.5 library
 - BB (Ip_solve) baseline solver

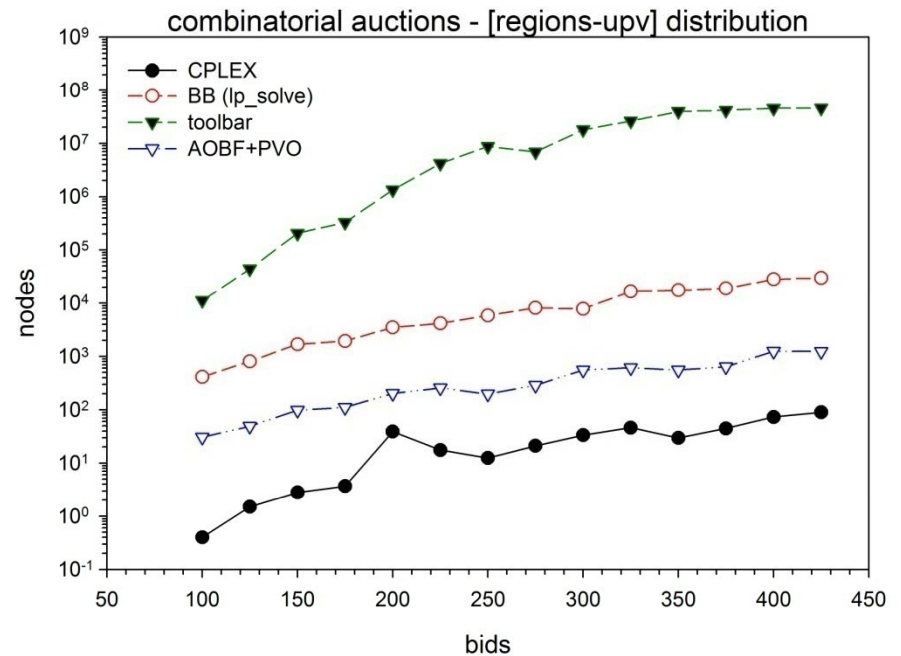
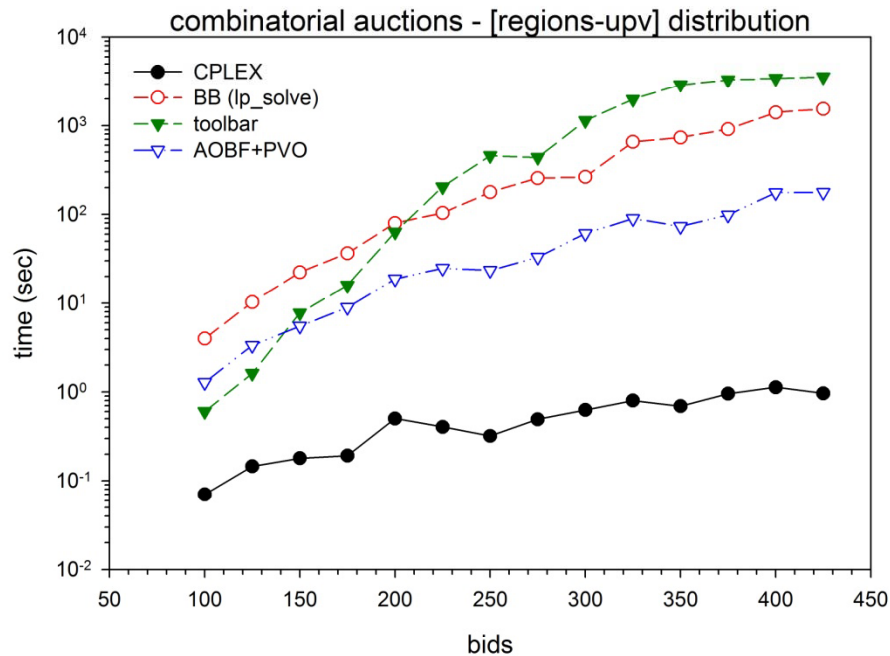
Combinatorial Auctions



Combinatorial auctions from **regions-upv** distribution with 100 goods and increasing number of bids. Time limit 1 hour.

Very large treewidth $\in [68, 184]$

Combinatorial Auctions



Combinatorial auctions from **regions-upv** distribution with 100 goods and increasing number of bids. Time limit 1 hour.

Very large treewidth $\in [68, 184]$

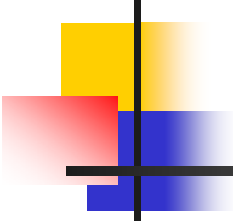
MAX-SAT Instances (pret)

Tree search Tree search Graph search

pret (w*, h)	BB CPLEX		AOBB AOBF		AOBB+PVO AOBF+PVO		AOBB-C AOBF-C	
	time	nodes	time	nodes	time	nodes	time	nodes
pret60-40 (6, 13)	- 676.94	- 3,926,422	7.88 7.56	1,255 1,202	8.41 8.70	1,216 1,326	7.38 3.58	1,216 568
pret60-60 (6, 13)	- 535.05	- 2,963,435	8.56 8.08	1,259 1,184	8.70 8.31	1,247 1,206	7.30 3.56	1,140 538
pret60-75 (6, 13)	- 402.53	- 2,005,738	6.97 7.38	1,124 1,145	6.80 8.42	1,089 1,149	6.34 3.08	1,067 506
pret150-40 (6, 15)	- out	-	95.11 101.78	6,625 6,535	108.84 101.97	7,152 6,246	75.19 19.70	5,625 1,379
pret150-60 (6, 15)	- out	-	98.88 106.36	6,851 6,723	112.64 102.28	7,347 6,375	78.25 19.75	5,813 1,393
pret150-75 (6, 15)	- out	-	108.14 98.95	7,311 6,282	115.16 103.03	7,452 6,394	84.97 20.95	6,114 1,430

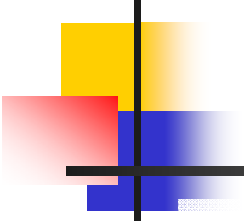
pret MAX-SAT instances. Time limit 10 hours.

BB solver could not solve any instance.



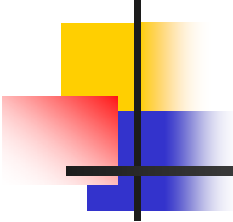
Summary

- New AND/OR search algorithms for 0-1 Integer Programming
- Dynamic variable orderings
- Superior to baseline OR Branch-and-Bound from the lp_solve library
- Outperform CPLEX on selected MAX-SAT instances (e.g., **pret, dubois**)



Outline

- Introduction
- Inference
- **Search**
 - Exact
 - AND/OR search trees
 - AND/OR Branch-and-Bound search
 - AND/OR search graphs (caching)
 - AND/OR search for 0-1 integer programming
 - **AND/OR search for multi-objective optimization**
 - Approximate: Sampling etc.
- Compilation: AND/OR Decision Diagrams
- Software



Multi-objective Constraint Optimization

- Many real-world problems involve multiple, conflicting and non-commensurate objectives
 - e.g., scheduling multi-orbit Earth observing satellites, risk-conscious combinatorial auctions, etc.
 - Find the set of equivalent or non-dominated costs of the set of feasible solutions (i.e., efficient or Pareto optimal frontier)
 - Current search methods (depth-first or best-first) are not sensitive to the **problem structure**
 - independent sub-problems are solved multiple times

Multi-objective Constraint Optimization

A	B	C	F ₁	A	B	D	F ₂	B	D	E	F ₃
0	0	0	5	0	0	0	1	0	0	0	1
0	0	1	2	0	0	1	4	0	0	1	3
0	1	0	3	0	1	0	0	0	1	0	5
0	1	1	4	0	1	1	2	0	1	1	4
1	0	0	2	1	0	0	6	1	0	0	1
1	0	1	3	1	0	1	5	1	0	1	3
1	1	0	0	1	1	0	6	1	1	0	5
1	1	1	2	1	1	1	5	1	1	1	4

(assume a cost vector representation of the functions)

$$f_1 = F_1(A, B, C) + F_2(A, B, D) + F_3(B, D, E)$$

(first objective)

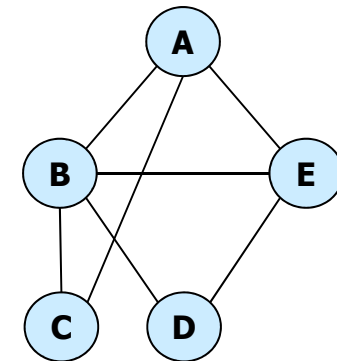
A	g ₁	B	g ₂	C	g ₃	D	g ₄	E	g ₅
0	0	0	0	0	0	0	0	0	0
1	1	1	2	1	3	1	4	1	5

$$f_2 = g_1(A) + g_2(B) + g_3(C) + g_4(D) + g_5(E)$$

(second objective)

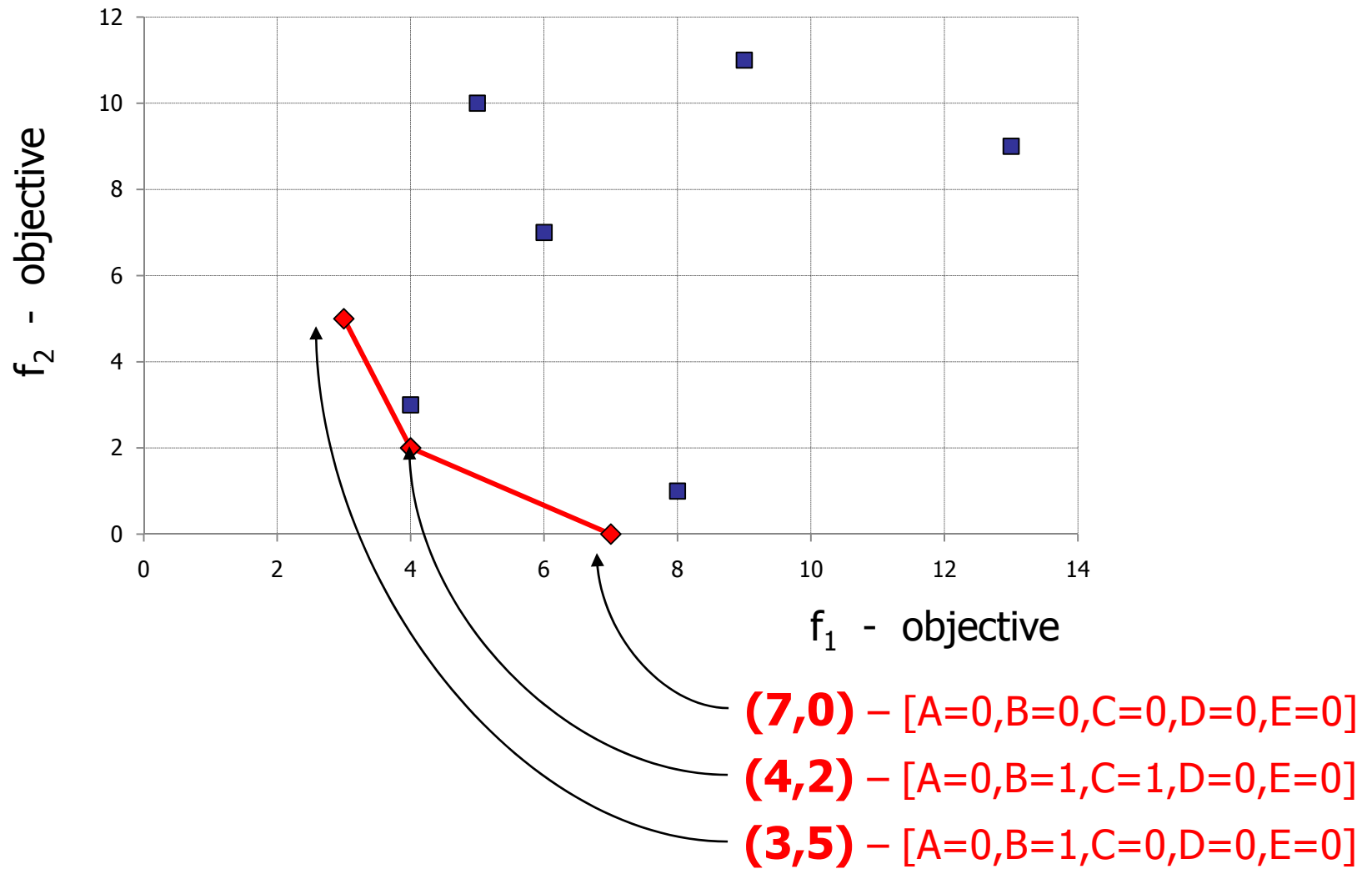
Task: minimize f_1 and f_2 simultaneously !

efficient frontier: $\{(3,5) (4,2) (7,0)\}$



constraint graph

Efficient frontier



Arc weights

A	B	C	F ₁	A	B	D	F ₂	B	D	E	F ₃
0	0	0	5	0	0	0	1	0	0	0	1
0	0	1	2	0	0	1	4	0	0	1	3
0	1	0	3	0	1	0	0	0	1	0	5
0	1	1	4	0	1	1	2	0	1	1	4
1	0	0	2	1	0	0	6	1	0	0	1
1	0	1	3	1	0	1	5	1	0	1	3
1	1	0	0	1	1	0	6	1	1	0	5
1	1	1	2	1	1	1	5	1	1	1	4

f₁ objective

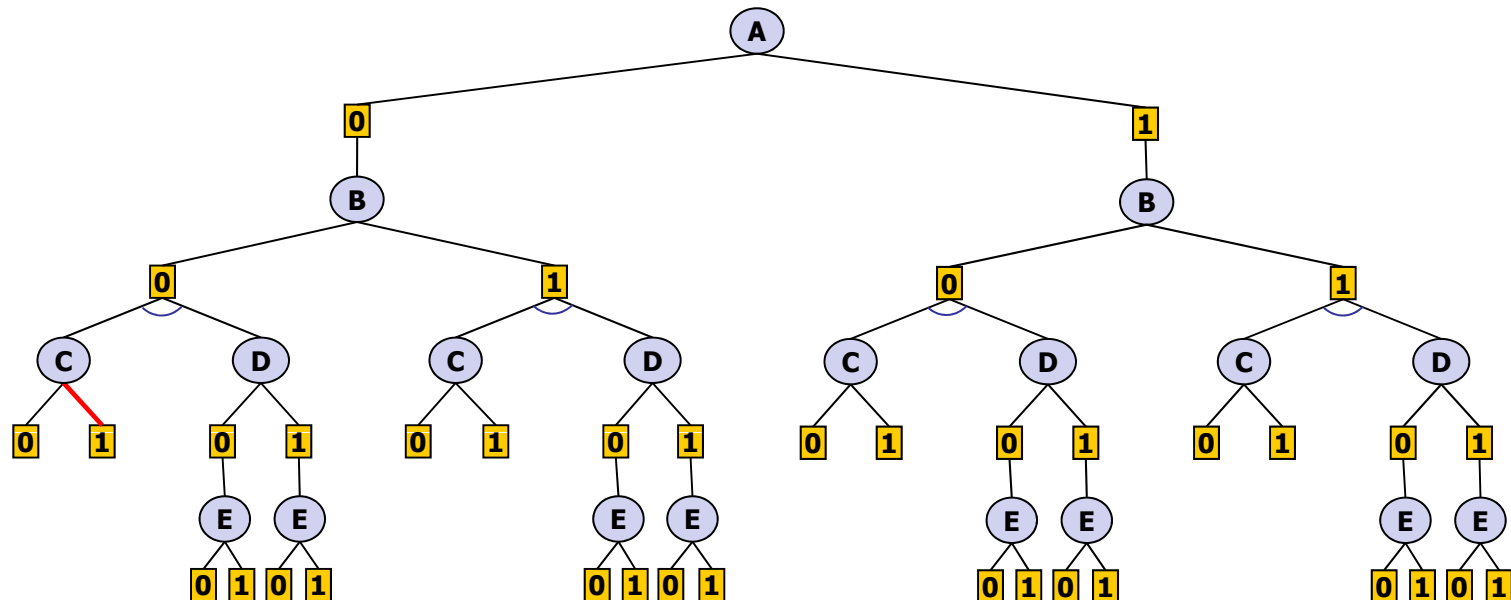
A	g ₁	B	g ₂	C	g ₃	D	g ₄	E	g ₅
0	0	0	0	0	0	0	0	0	0
1	1	1	2	1	3	1	4	1	5

f₂ objective

$$w(C,1) = (F_1(a,b,1), g_3(1)) = (2,3)$$

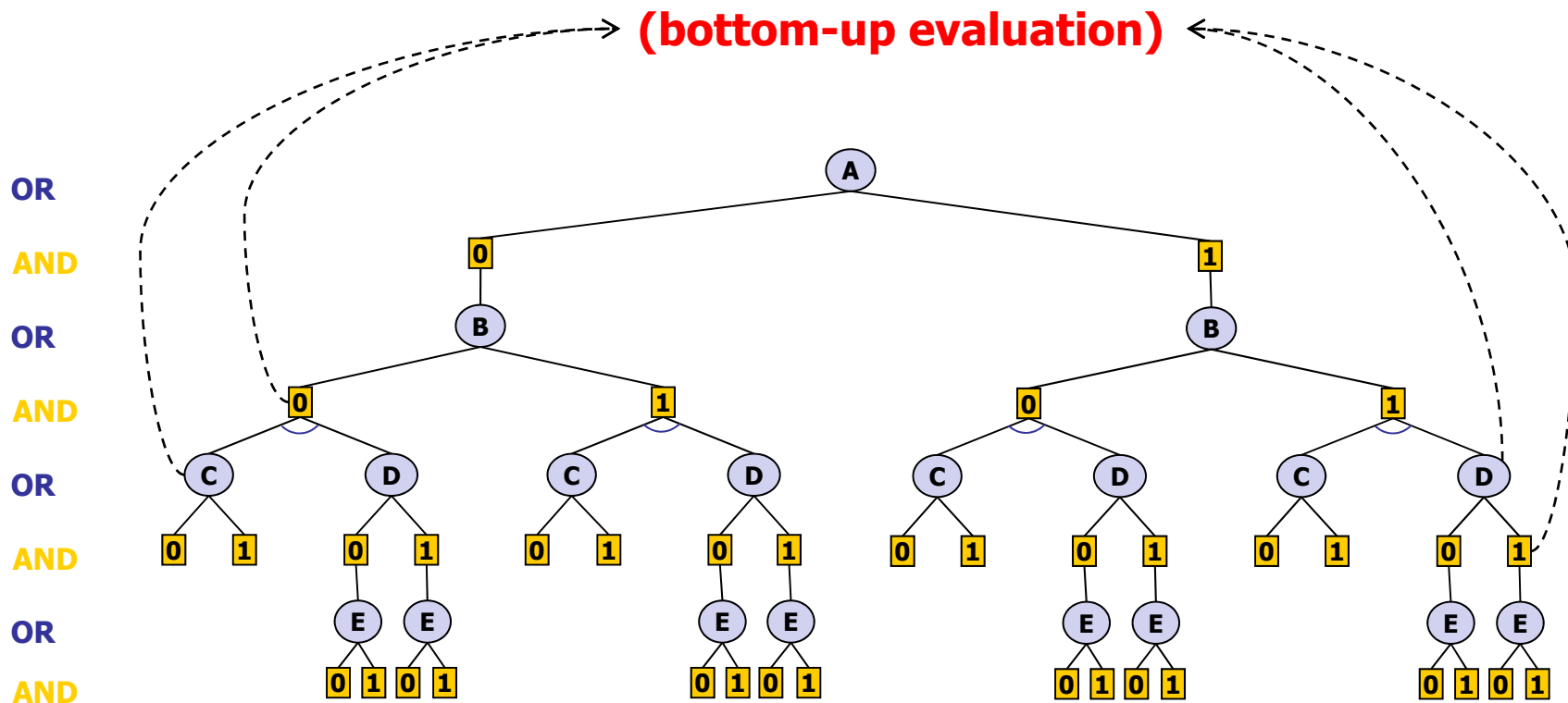
weights computed based on input functions

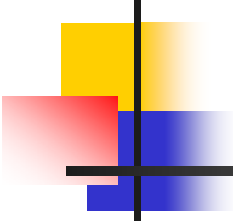
OR
AND
OR
AND
OR
AND



Node values

$$v(n) = \begin{cases} (0, \dots, 0), & \text{if } n \text{ is terminal AND} \\ \sum_{n' \in \text{succ}(n)} v(n'), & \text{if } n \text{ is AND} \\ || \{w(n, n') + v(n') \mid n' \in \text{succ}(n)\} ||, & \text{if } n \text{ is OR} \end{cases}$$





Experiments for MO-COPs

■ Algorithms

■ **MO-AOBB+SMB(i)** (Marinescu, CP 2009)

- AND/OR Branch-and-Bound with mini-bucket heuristics

■ **MO-BB+SMB(i)**

- OR Branch-and-Bound with mini-bucket heuristics

■ **MO-RDS** (Rollon & Larrosa, AAI 2007)

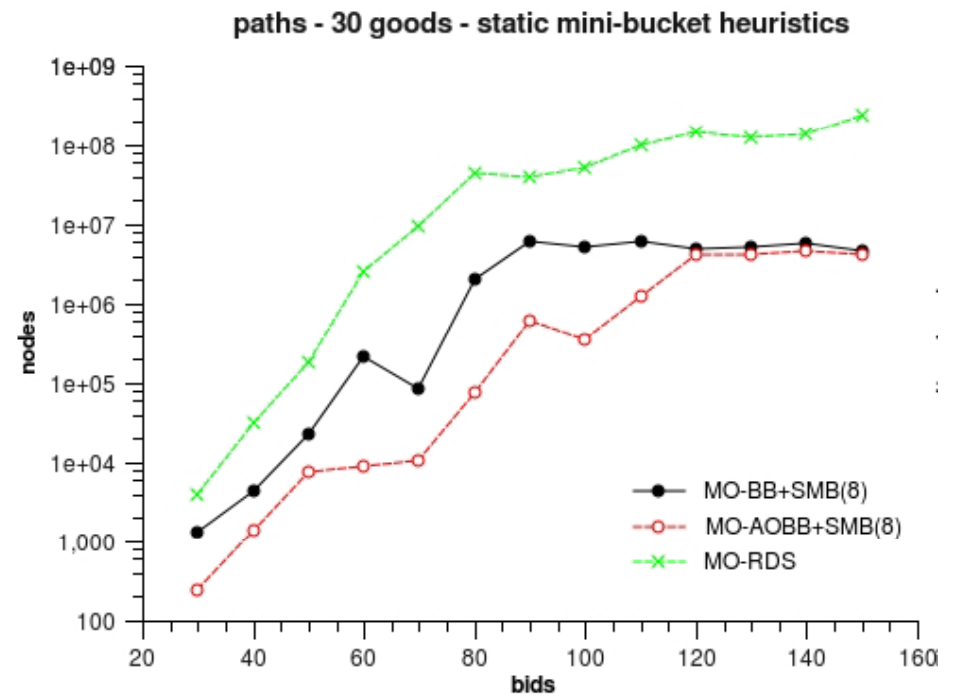
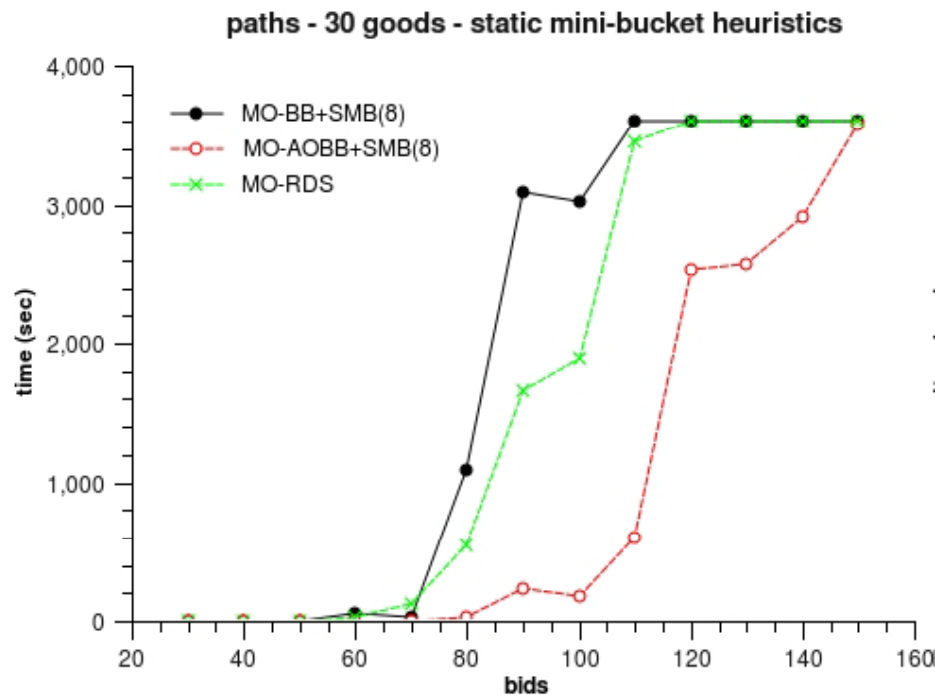
- Multi-objective Russian Doll Search

■ Benchmarks

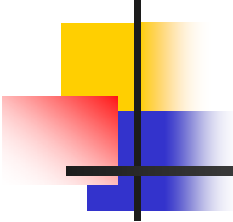
■ Bi-objective combinatorial auctions

■ Bi-objective WCSPs derived from classic WCSPs

Bi-objective Combinatorial Auctions



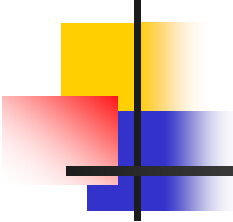
Time limit 1 hour. Static mini-bucket heuristics.
Each data point is the average over 20 random instances.



Bi-objective WCSPs

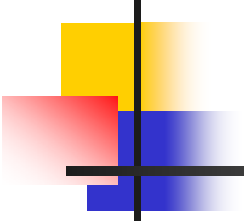
instance (n,c,k)	(w*,h) E	MO-RDS	MO-BB+SMB(14)		MO-BB+SMB(20)	
			MO-AOBB+SMB(14)		MO-AOBB+SMB(20)	
			time	nodes	time	nodes
c432 (432,648,2)	(27,35) 31	-	- 38.99	- 96K	- 39.95	- 50K
s1488 (667,1000,2)	(47,67) 10	-	- -	- -	- 1751.86	- 9M
90-16-1 (256,512,2)	(24,82) 8	-	- 328.61	- 1M	- 84.88	- 127K
bm05-03 (1005,2010,2)	(22,59) 2	-	- 7.59	- 11K	- 65.96	- 20K
mm-03-08-03 (1220,2440,2)	(20,57) 1	-	- 10.50	- 47K	- 105.90	- 6K

Time limit 10 hours.



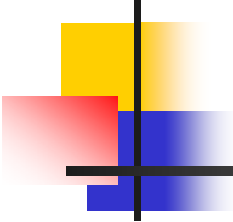
Algorithms for AND/OR Space

- **Back-jumping** for CSPs
(Gaschnig 1977), (Dechter 1990), (Prosser, Bayardo & Mirankar, 1995)
- **Pseudo-search re-arrangement**, for any CSP task
(Freuder & Quinn 1985)
- **Pseudo-tree search for soft constraints**
(Larrosa, Meseguer & Sanchez, 2002)
- **Recursive Conditioning**
(Darwiche, 2001), explores the AND/OR tree or graph for any query
- **BTD: Searching tree-decompositions** for optimization
(Jeagou & Terrioux, 2004)
- **Value Elimination**
(Bacchus, Dalmao & Pittasi, 2003)



Outline

- Introduction
- Inference
- **Search**
 - Exact
 - **Approximate: Sampling**
- Compilation: AND/OR Decision Diagrams
- Software



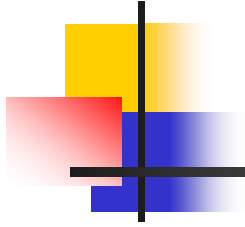
Sampling: Approximation of Search

1. Importance Sampling
2. Markov Chain Monte Carlo: Gibbs Sampling
3. Sampling in presence of Determinism
4. Rao-Blackwellisation
5. AND/OR importance sampling

See :Sampling Techniques for Probabilistic and
Deterministic Graphical models [PDF](#)

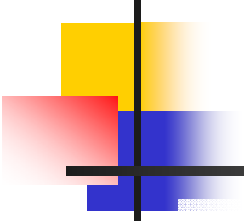
Tutorial, AAAI 2010, Atlanta, GA, July 12, 2010:

<http://www.ics.uci.edu/~dechter/talks.html>



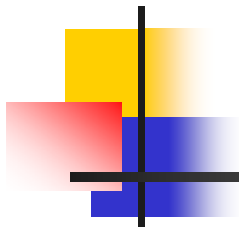
Outline

- Introduction
- Inference
- Search
- **Compilation: AND/OR Decision Diagrams**
- Software



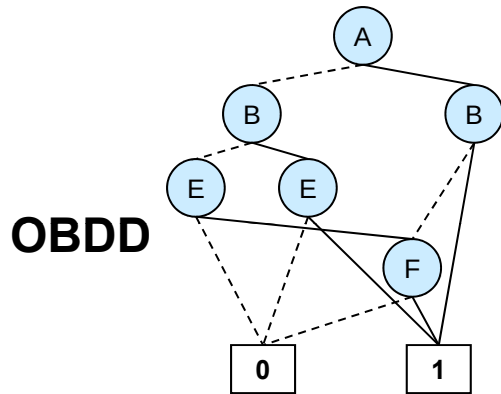
Outline

- Introduction
- Inference
- Search
- **Compilation**
 - AND/OR Decision Diagrams
 - Apply Operator
 - Bottom up (Variable elimination)
 - Top down (AND/OR search)
- Software

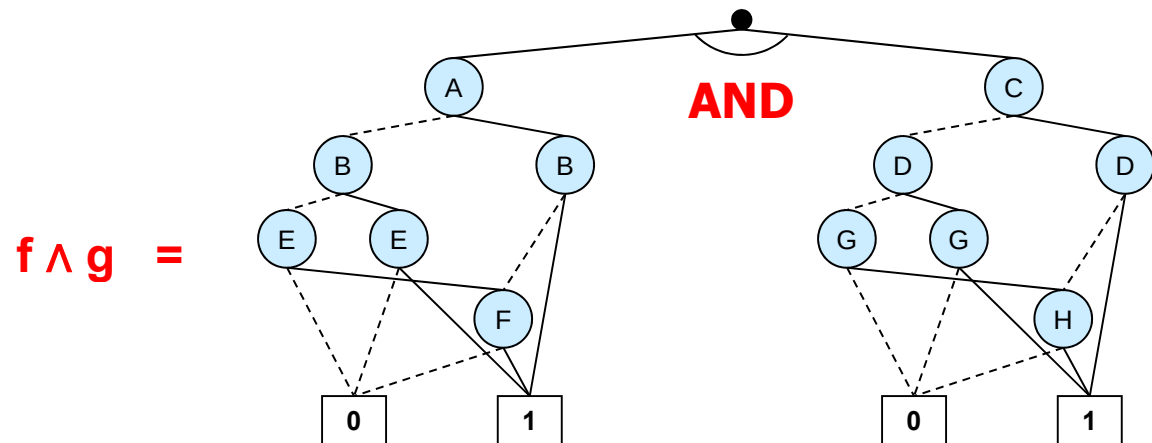
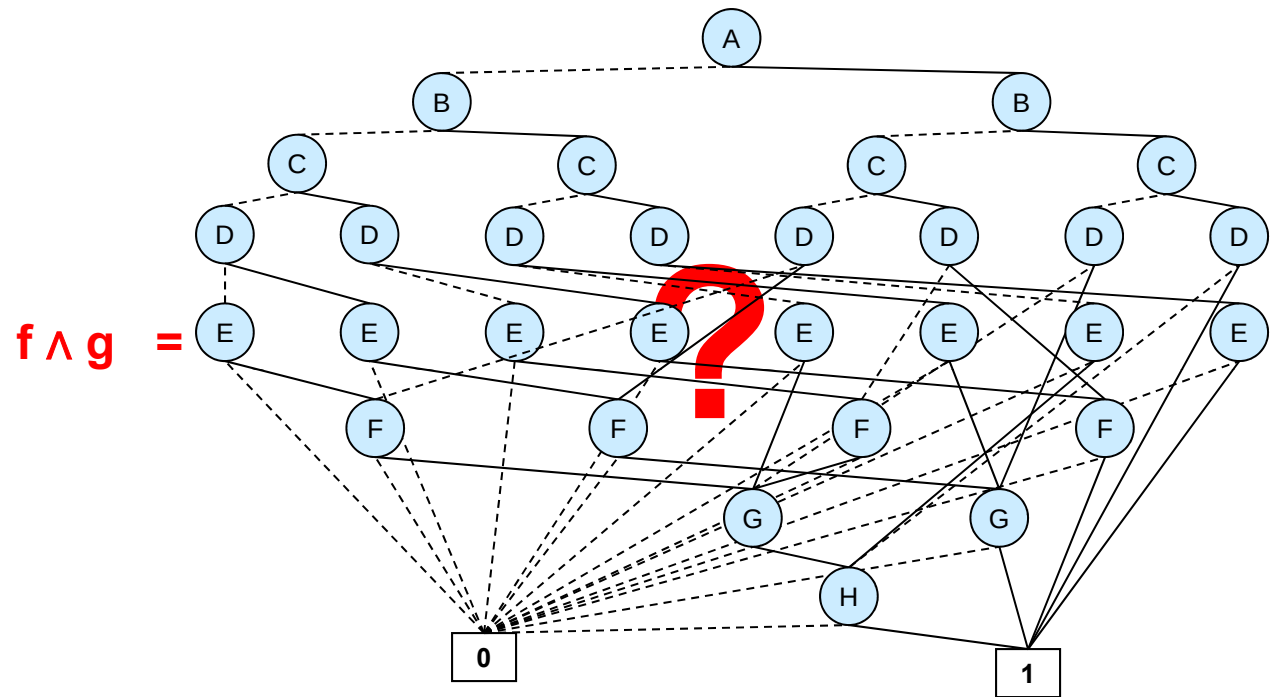
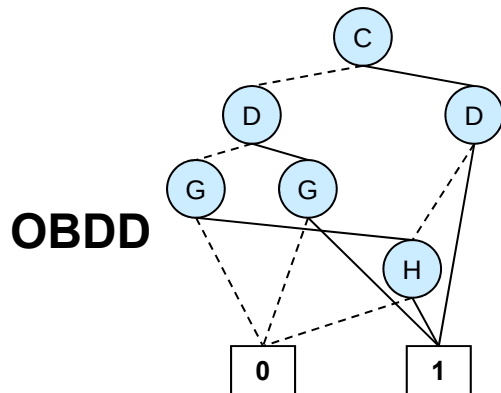


Exploiting Structure

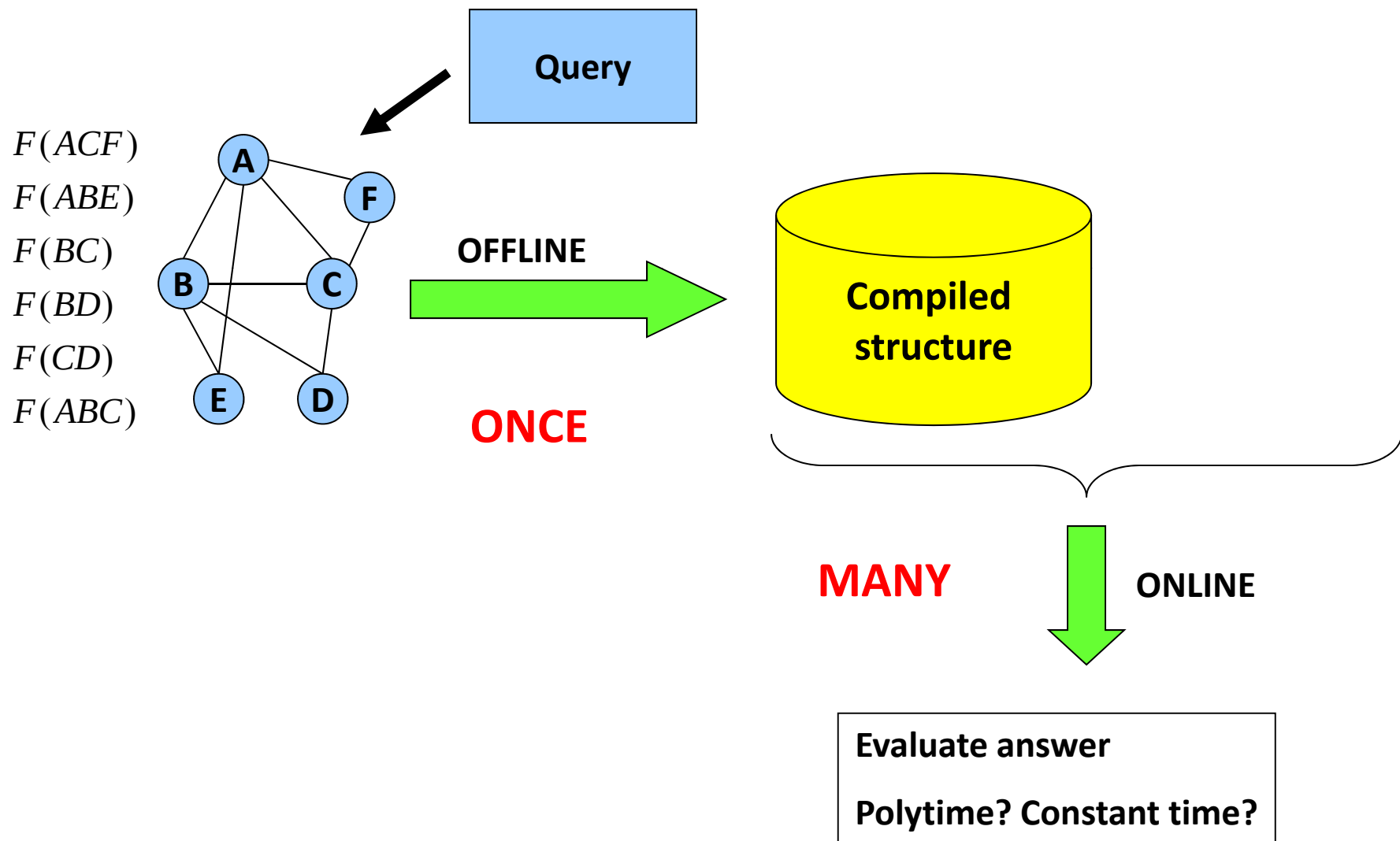
$$f = (A \vee E) \wedge (B \vee F)$$



$$g = (C \vee G) \wedge (D \vee H)$$



Compilation of Graphical Models

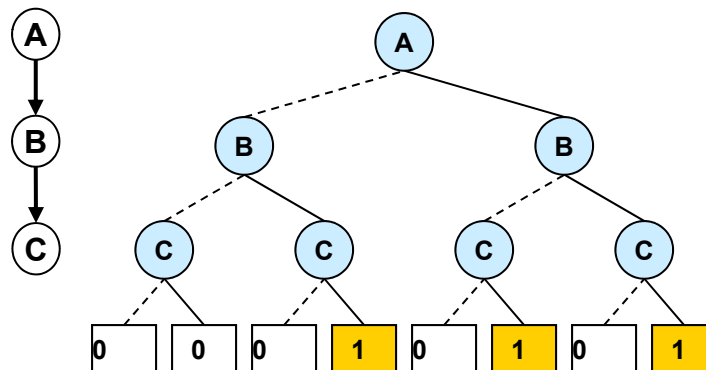


Ordered Binary Decision Diagram

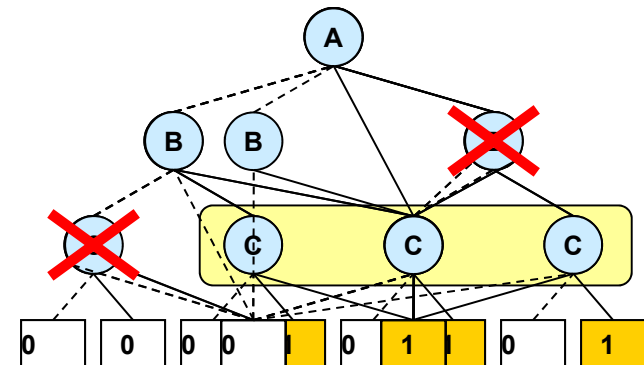
$$B = \{0,1\} \quad f : B^3 \rightarrow B$$

A	B	C	f(ABC)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Table



Decision tree

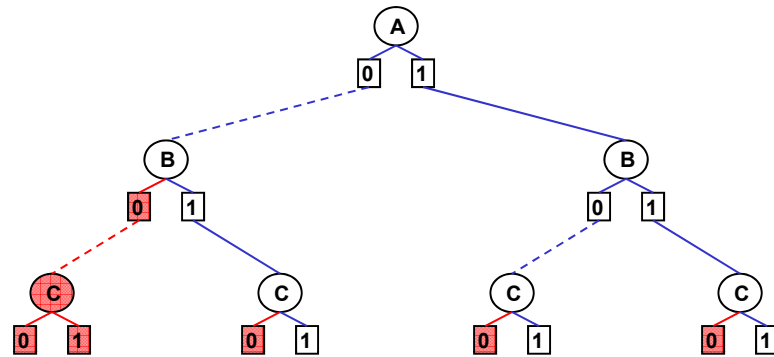
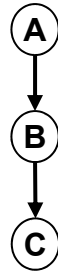


- 1) Merge identical children
[Bryant86]
- 2) Remove redundant nodes

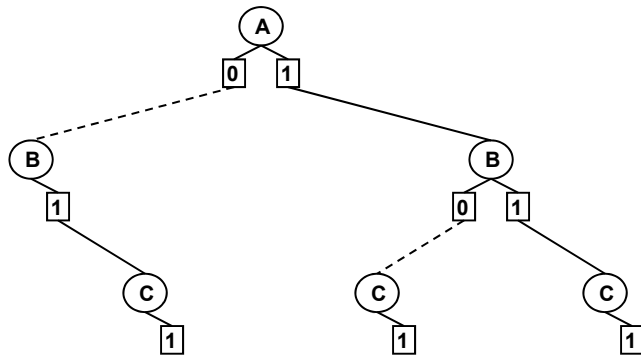
Ordering enables efficient operations

Minimal AND/OR Graphs (MAO)

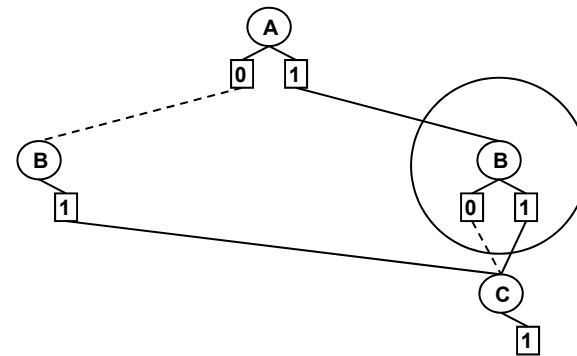
A	B	C	f(ABC)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



Full AND/OR search tree



Backtrack free AND/OR search tree



redundant

Minimal AND/OR search graph

Minimal AND/OR graph = closure under "merge"
 = unique fix point of "merge"

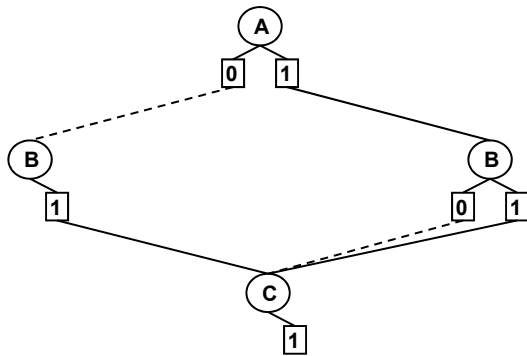
From MAO to Decision Diagram

A	B	C	f(ABC)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

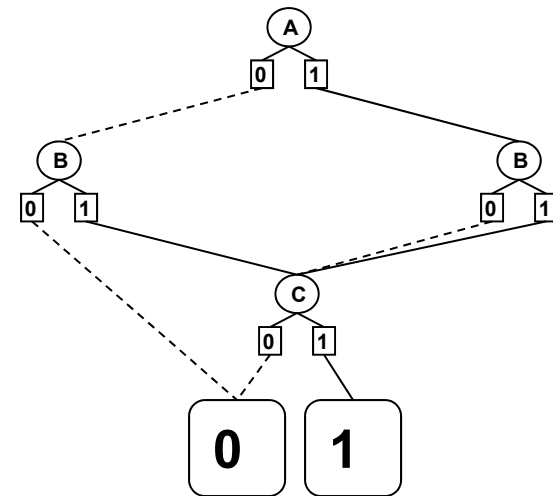
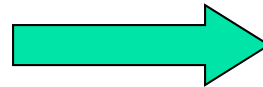


Point dead-ends to terminal node “0”

Point goods to terminal node “1”



Minimal AND/OR graph



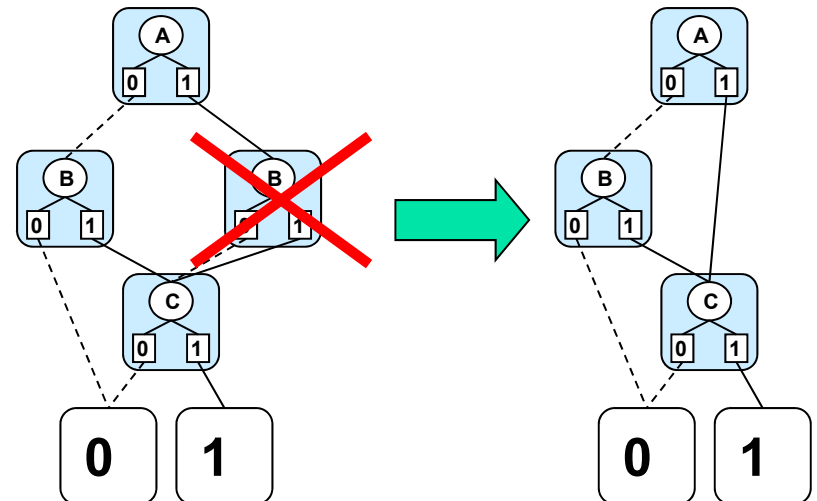
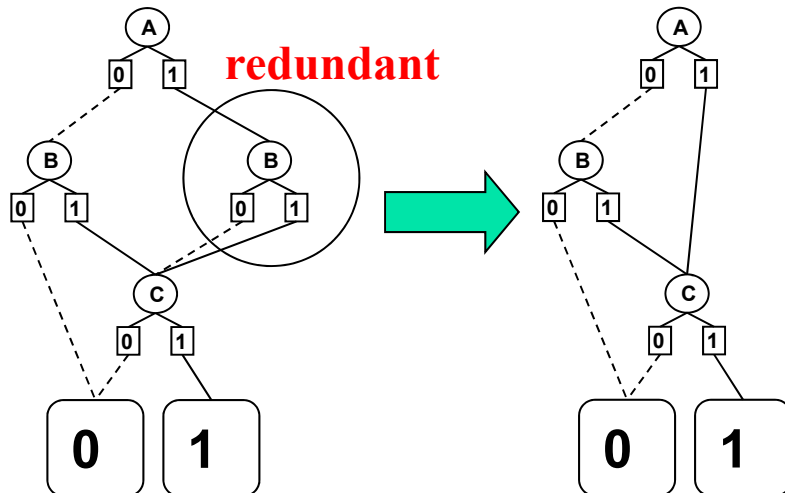
Decision Diagram

Removing Redundancy

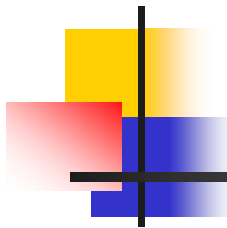
A	B	C	f(ABC)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



Group OR node together with its AND children into a meta-node



OBDD
(pseudo tree is a **chain**)

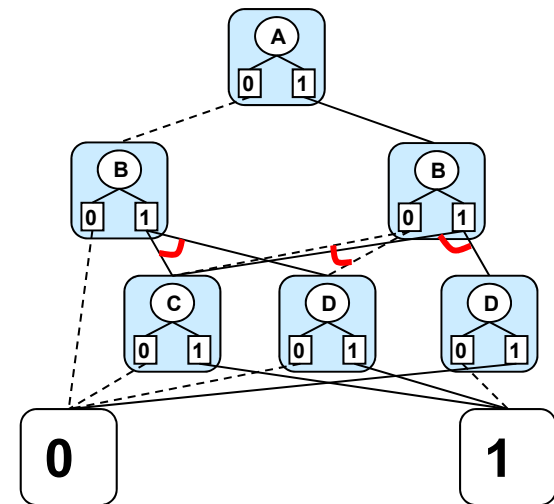
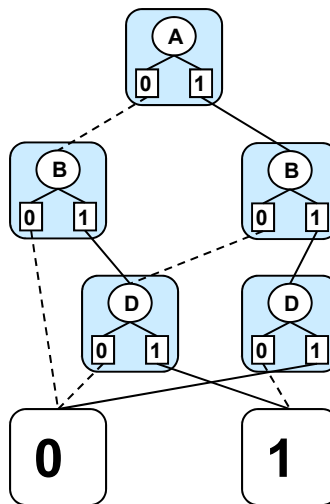
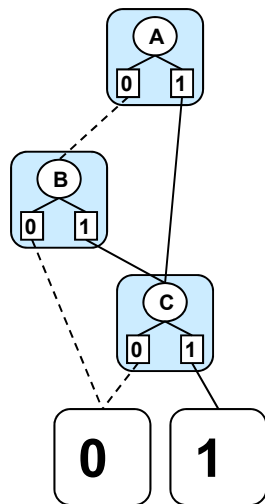
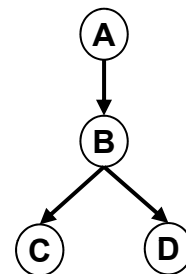


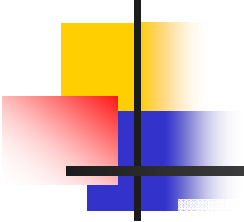
AOBDD

A	B	C	f(ABC)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



A	B	D	g(ABD)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

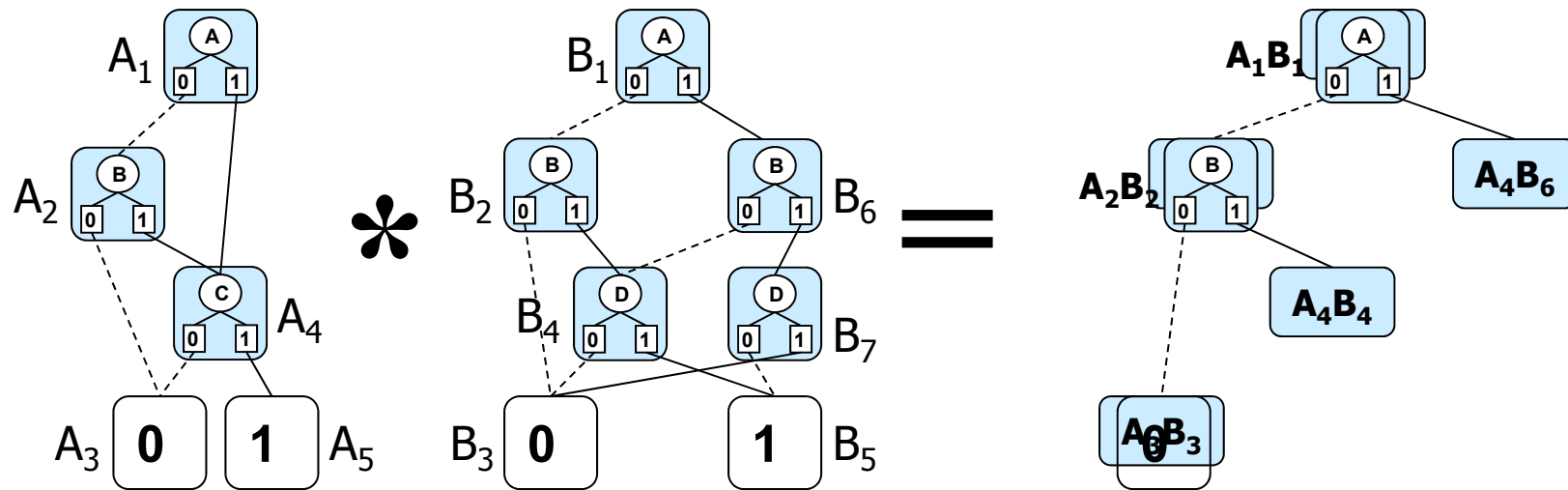
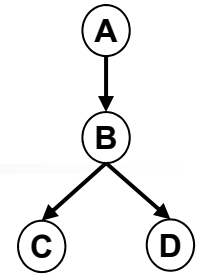




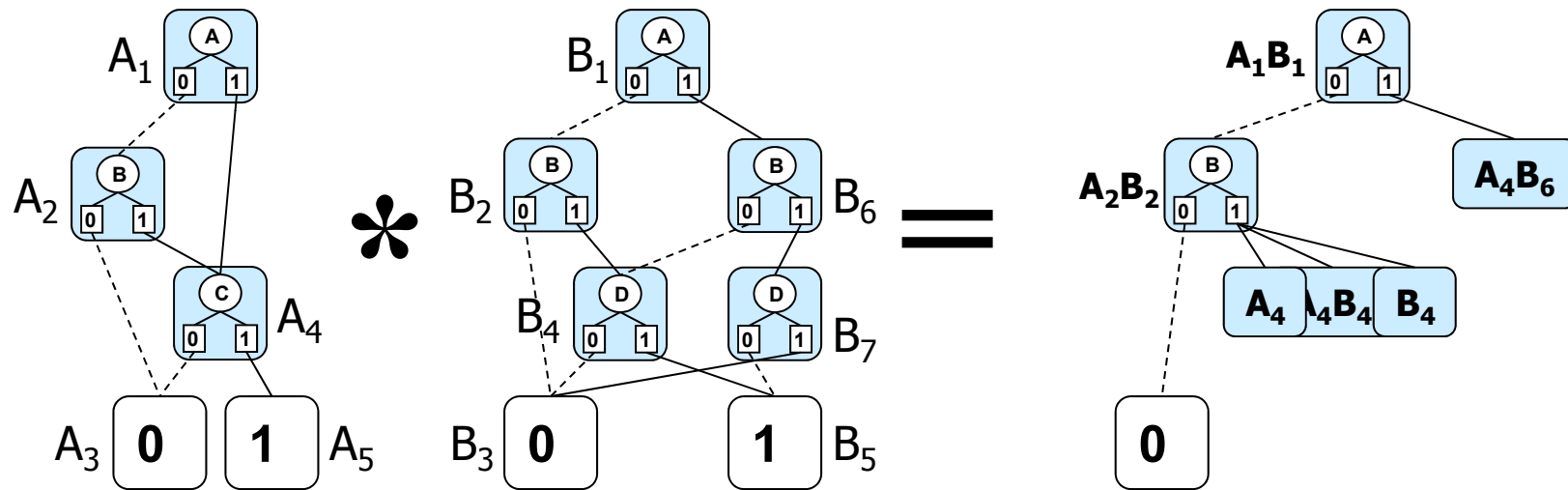
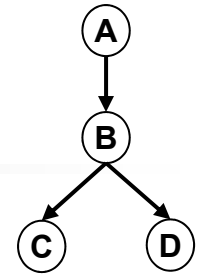
Outline

- Introduction
- Inference
- Search
- **Compilation**
 - AND/OR Decision Diagrams
 - **Apply Operator**
 - Bottom up (Variable elimination)
 - Top down (AND/OR search)
- Software

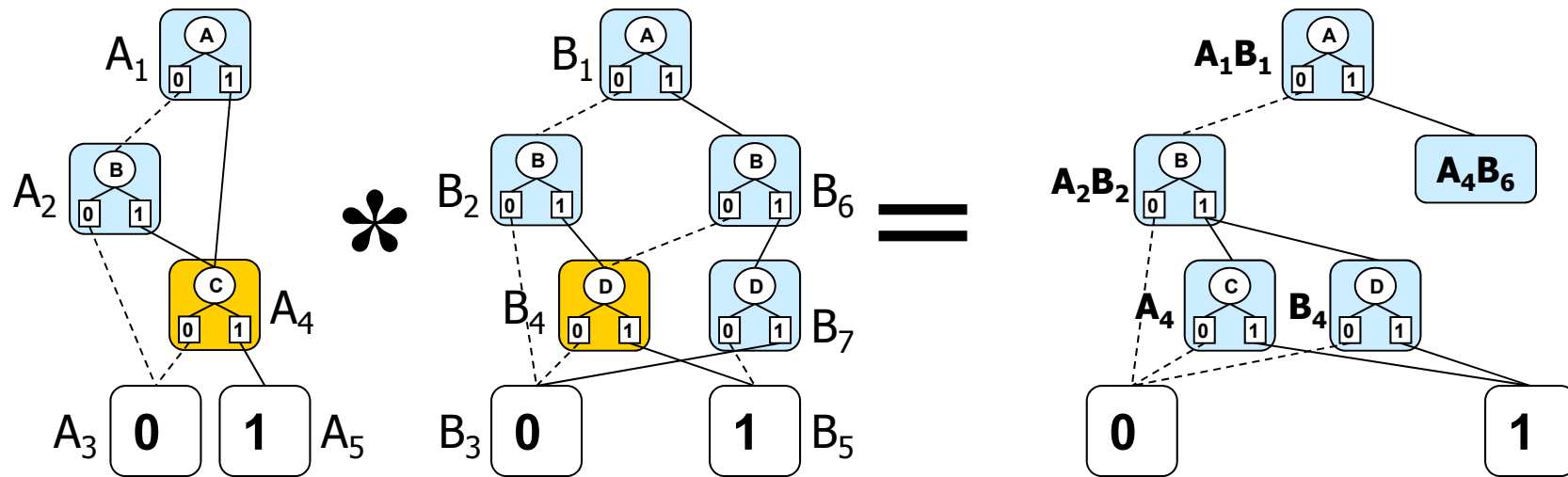
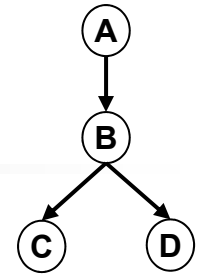
Apply Operator



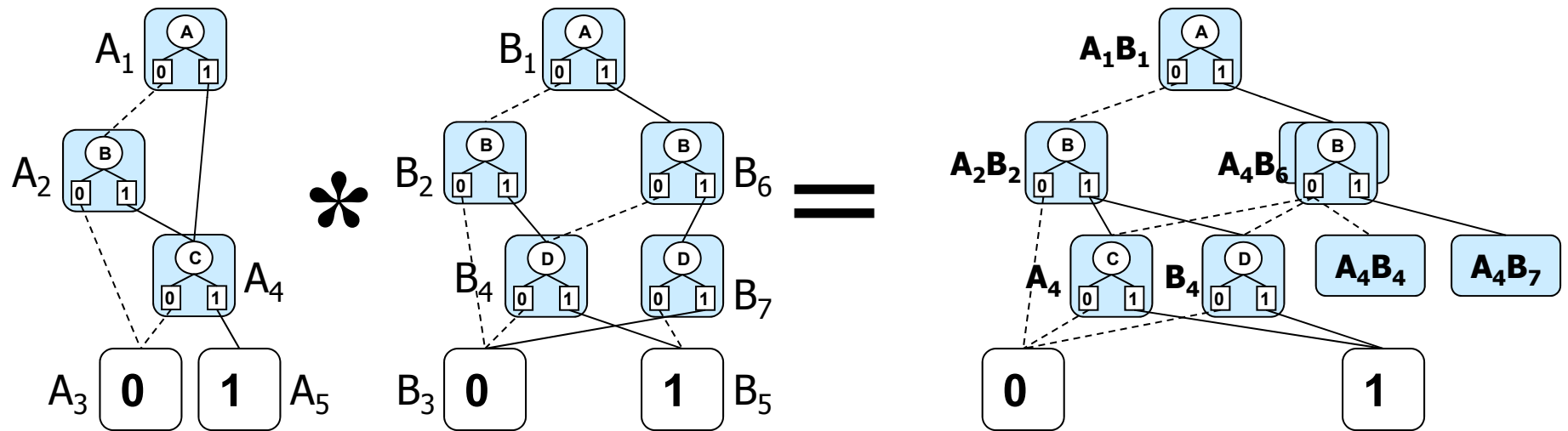
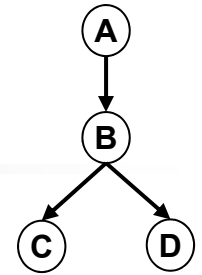
Apply Operator



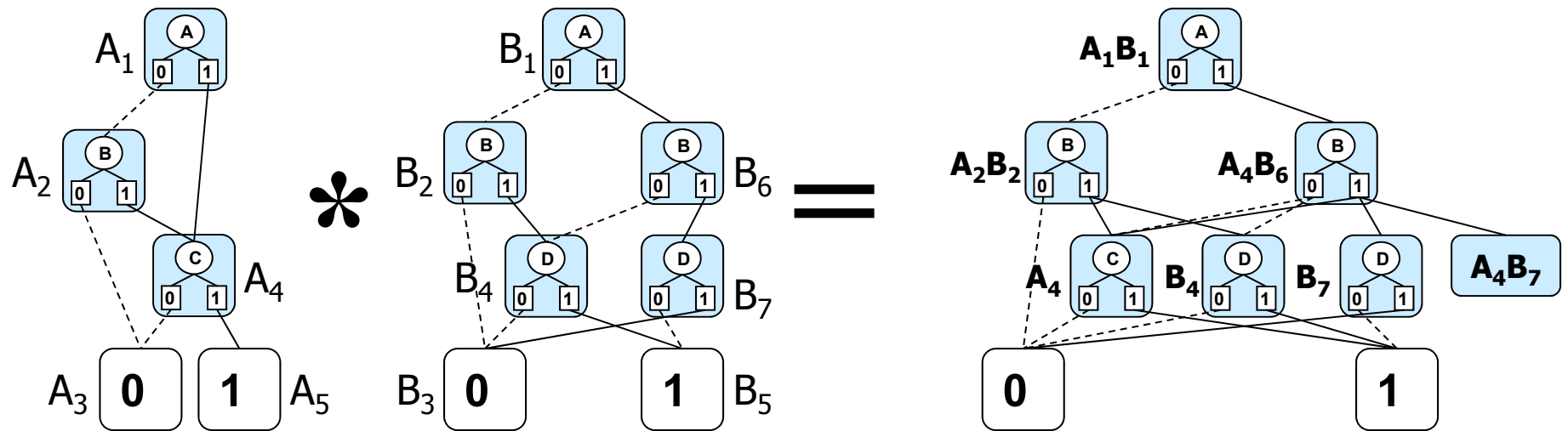
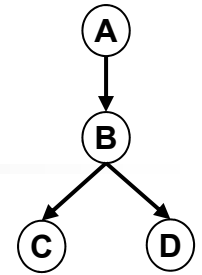
Apply Operator

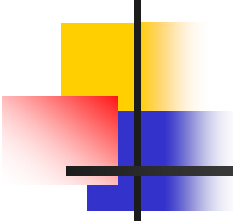


Apply Operator



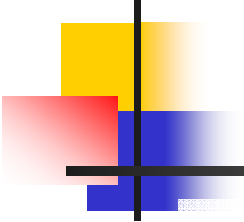
Apply Operator





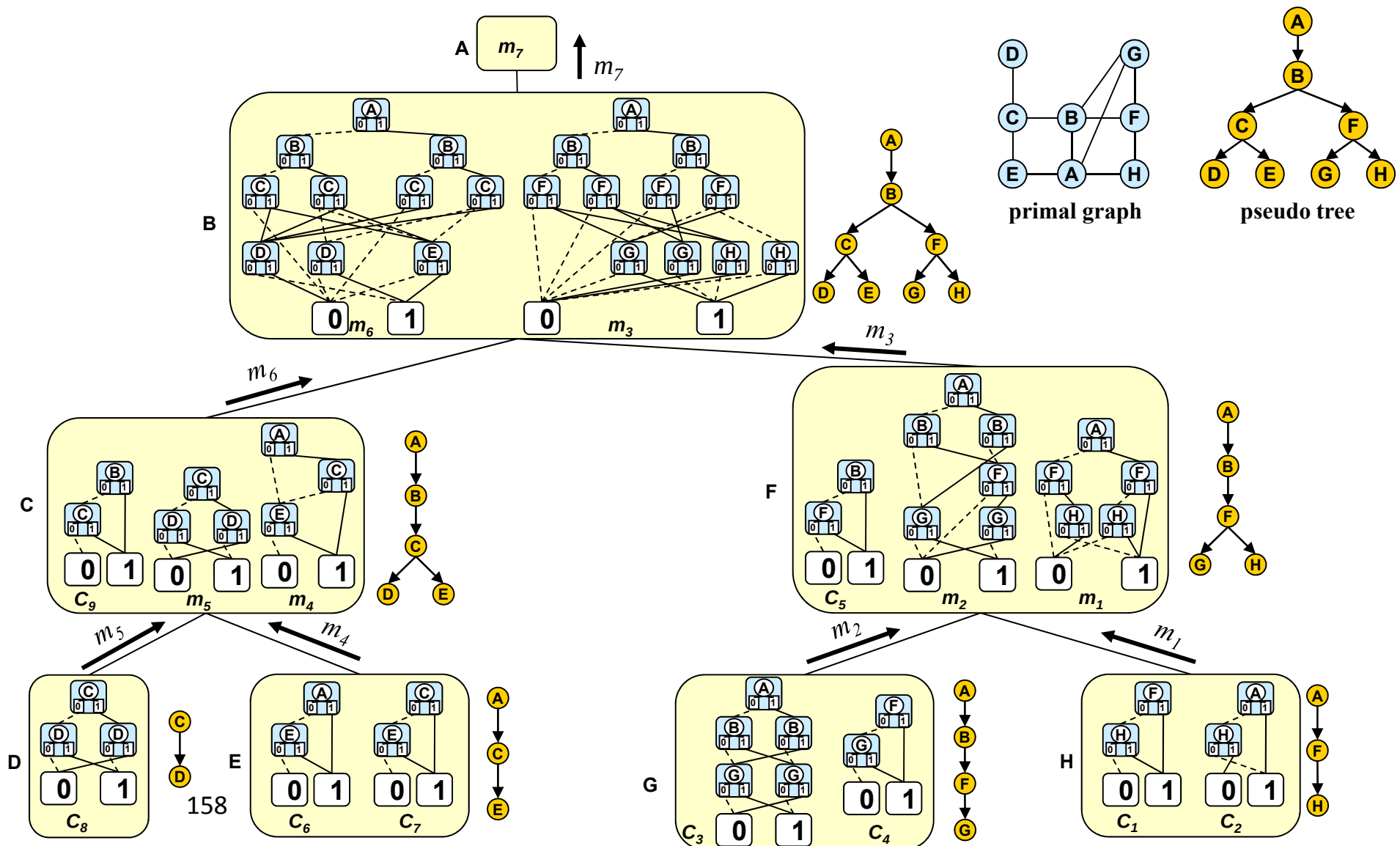
And/Or Multi-Valued Decision Diagrams

- AOMDDs are:
 - AND/OR search graphs
 - **canonical representations**, given a pseudo tree
 - Defined by two rules:
 - All isomorphic subgraphs are merged
 - There are no redundant (meta) nodes

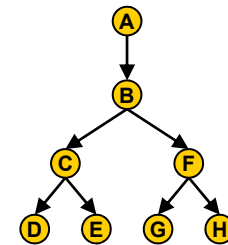
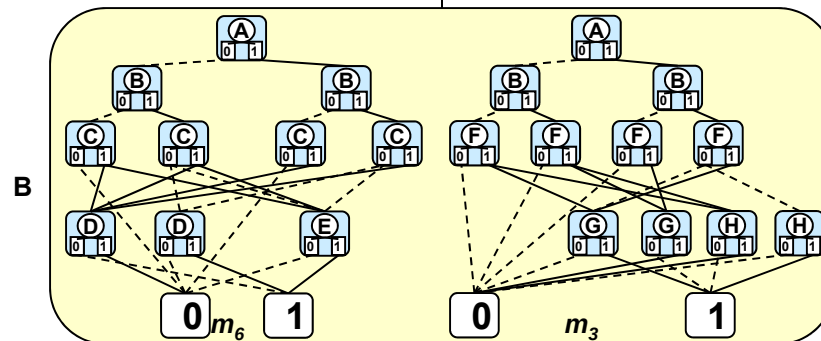
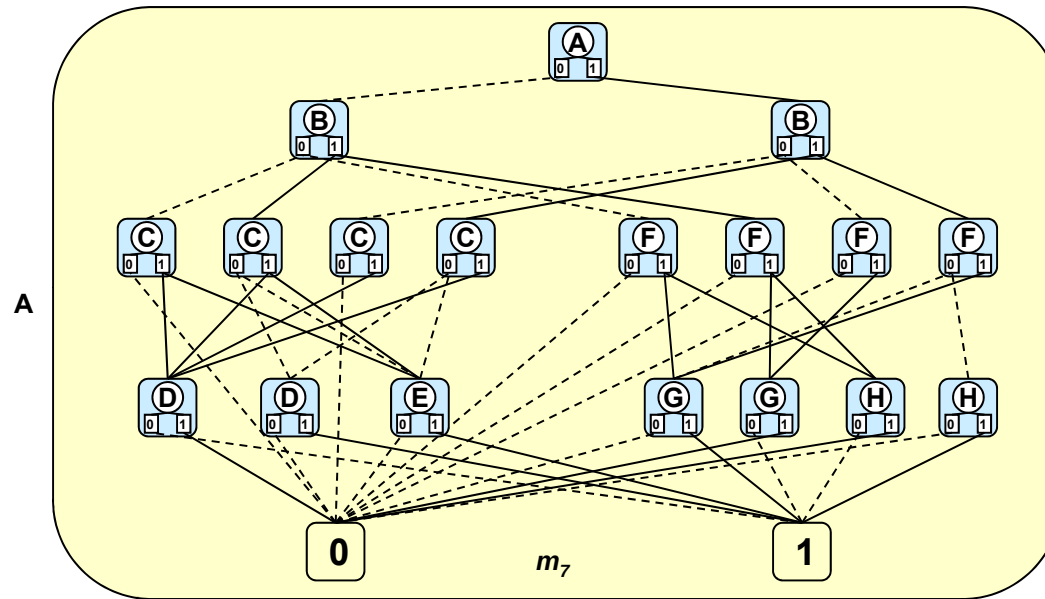


Outline

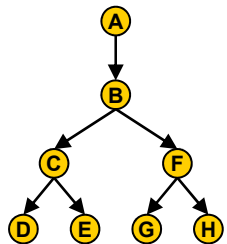
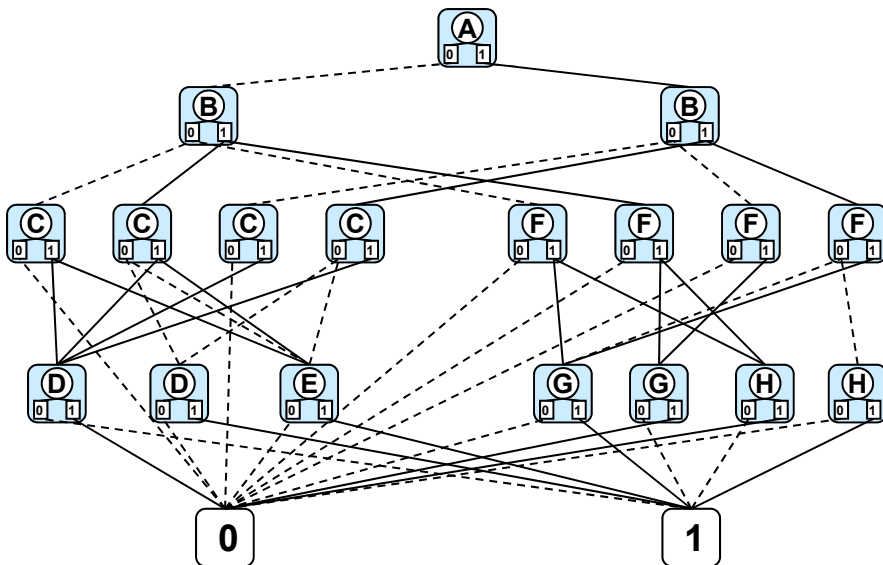
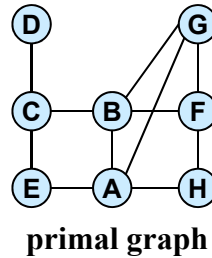
- Introduction
- Inference
- Search
- **Compilation**
 - AND/OR Decision Diagrams
 - Apply Operator
 - **Bottom up (Variable elimination)**
 - Top down (AND/OR search)
- Software

$$(f \vee h) \wedge (a \vee !h) \wedge (a \# b \# g) \wedge (f \vee g) \wedge (b \vee f) \wedge (a \vee e) \wedge (c \vee e) \wedge (c \# d) \wedge (b \vee c)$$


Example (continued)



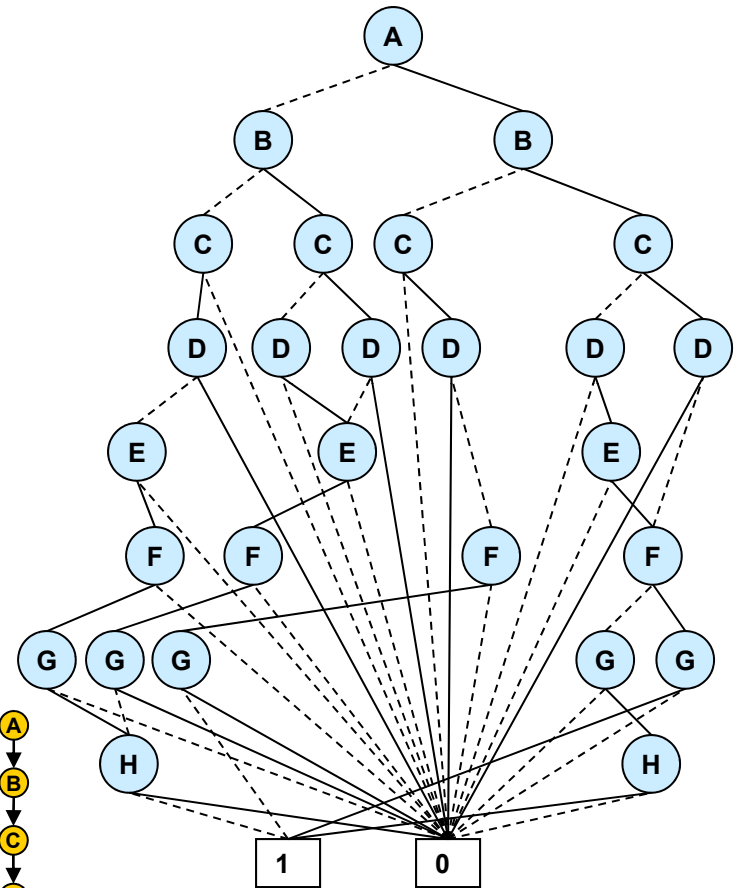
AOBDD vs. OBDD



AOBDD

18 nonterminals

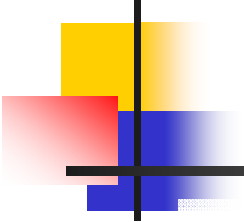
47 arcs



OBDD

27 nonterminals

54 arcs



Outline

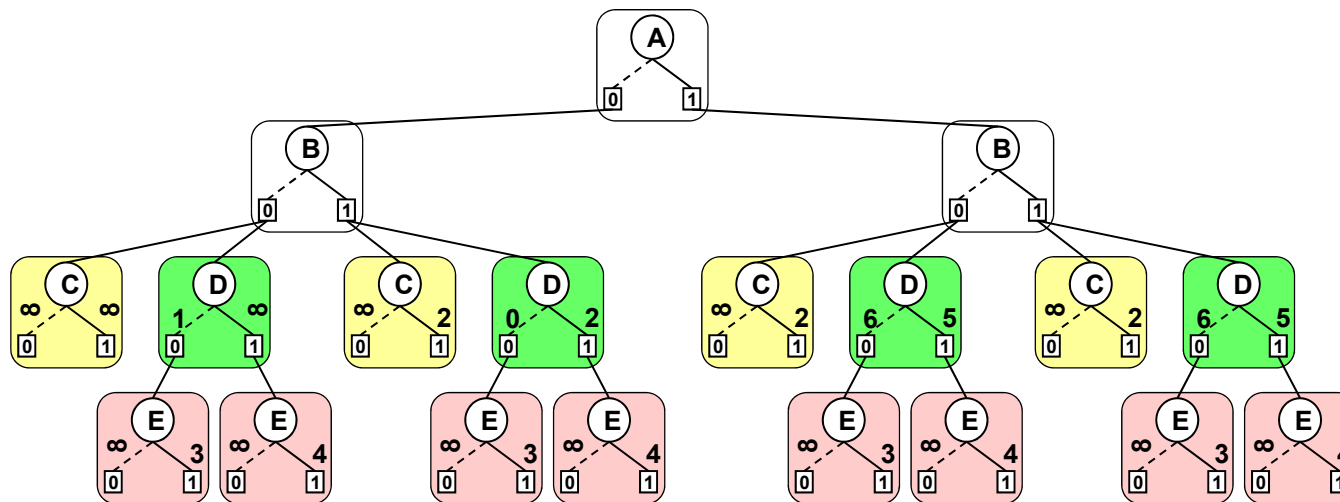
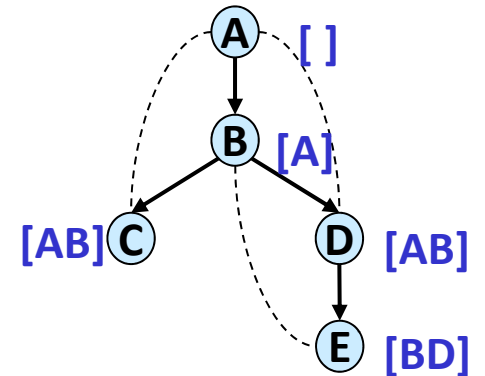
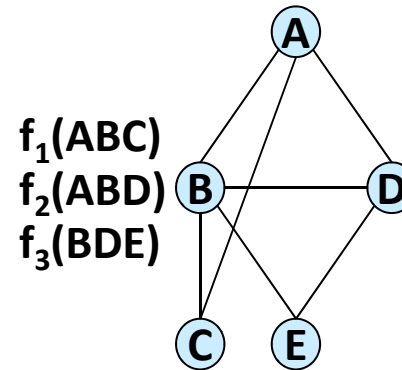
- Introduction
- Inference
- Search
- **Compilation**
 - AND/OR Decision Diagrams
 - Apply Operator
 - Bottom up (Variable elimination)
 - **Top down (AND/OR search)**
- Software

Constraint Optimization - AND/OR Tree

A	B	C	$f_1(ABC)$
0	0	0	∞
0	0	1	∞
0	1	0	∞
0	1	1	2
1	0	0	∞
1	0	1	2
1	1	0	∞
1	1	1	2

A	B	D	$f_2(ABD)$
0	0	0	1
0	0	1	∞
0	1	0	0
0	1	1	2
1	0	0	6
1	0	1	5
1	1	0	6
1	1	1	5

B	D	E	$f_3(BDE)$
0	0	0	∞
0	0	1	3
0	1	0	∞
0	1	1	4
1	0	0	∞
1	0	1	3
1	1	0	∞
1	1	1	4

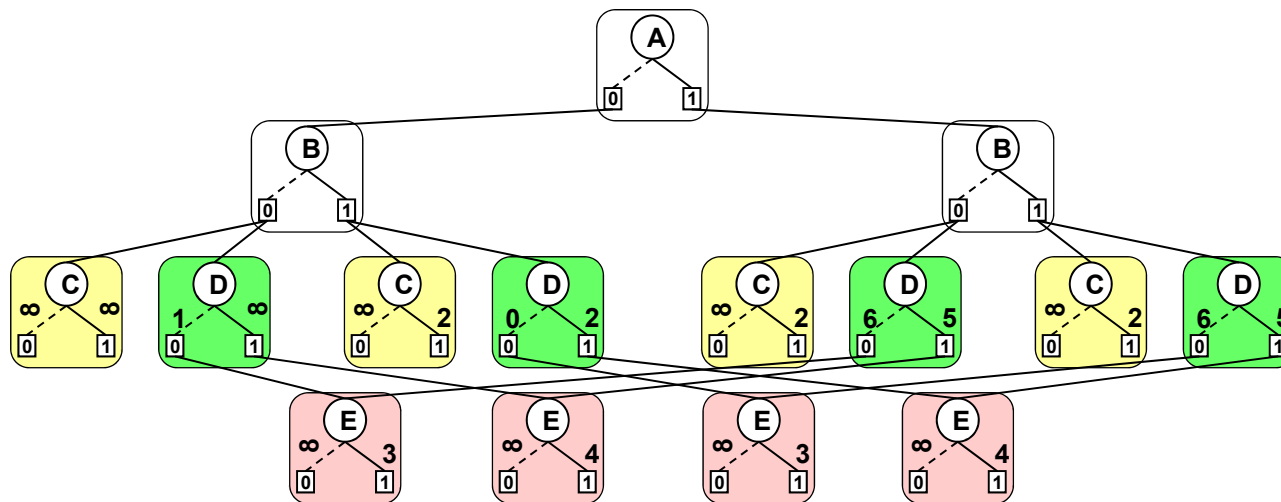
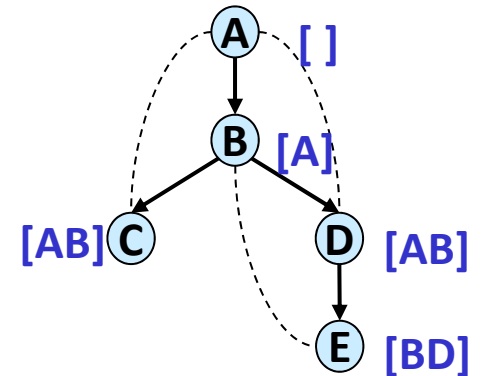
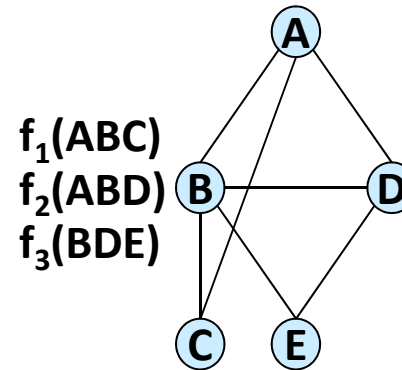


AND/OR Context Minimal Graph

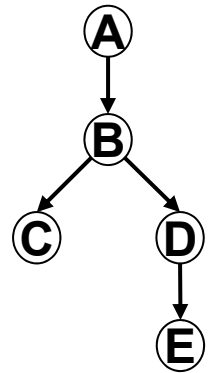
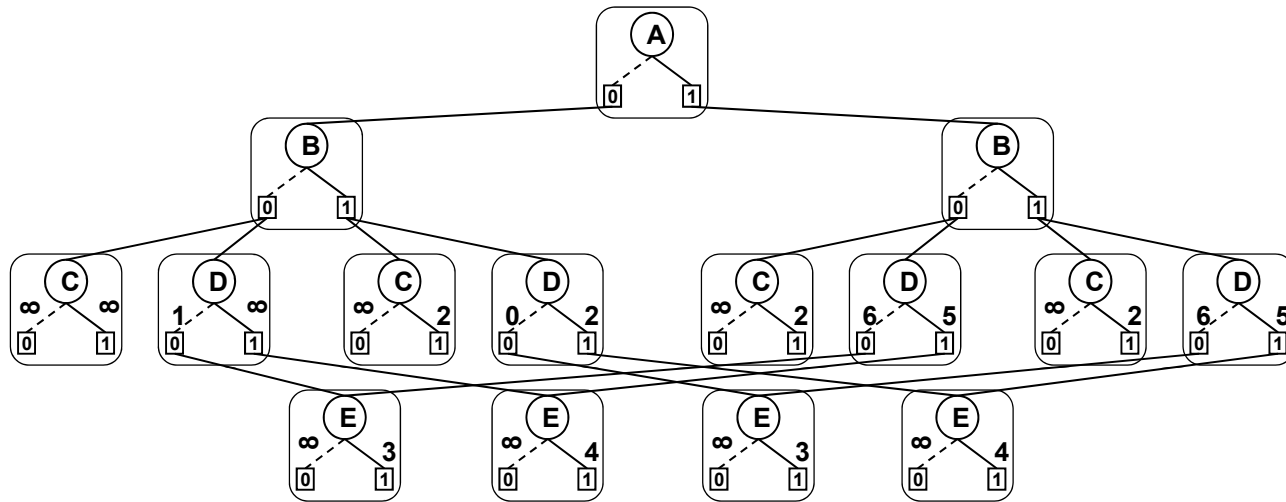
A	B	C	$f_1(ABC)$
0	0	0	∞
0	0	1	∞
0	1	0	∞
0	1	1	2
1	0	0	∞
1	0	1	2
1	1	0	∞
1	1	1	2

A	B	D	$f_2(ABD)$
0	0	0	1
0	0	1	∞
0	1	0	0
0	1	1	2
1	0	0	6
1	0	1	5
1	1	0	6
1	1	1	5

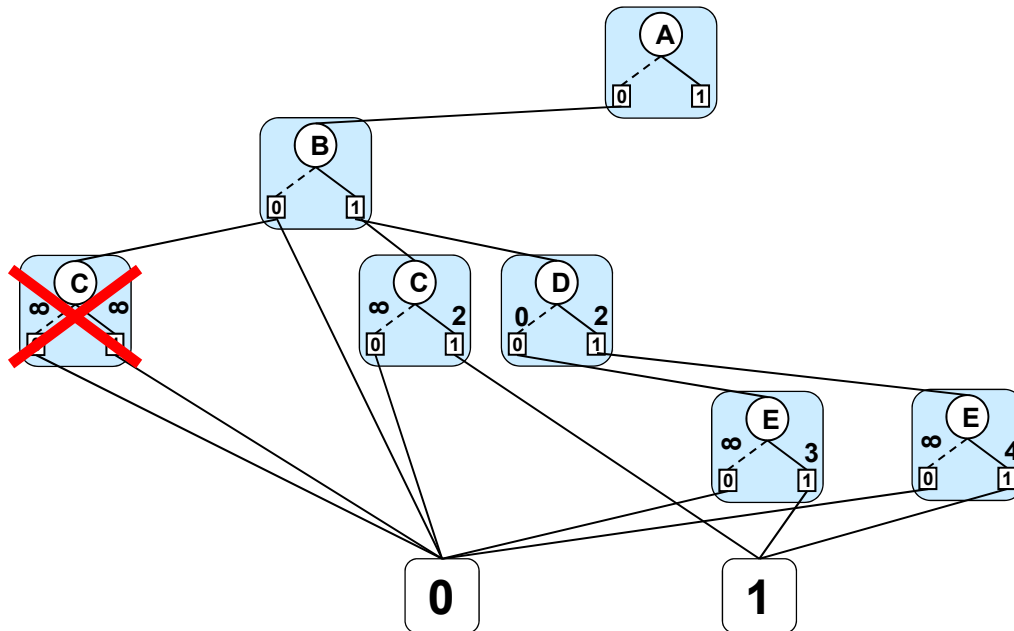
B	D	E	$f_3(BDE)$
0	0	0	∞
0	0	1	3
0	1	0	∞
0	1	1	4
1	0	0	∞
1	0	1	3
1	1	0	∞
1	1	1	4



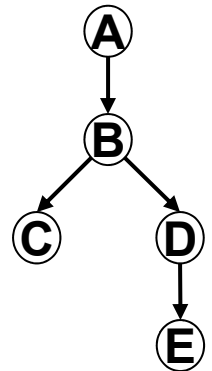
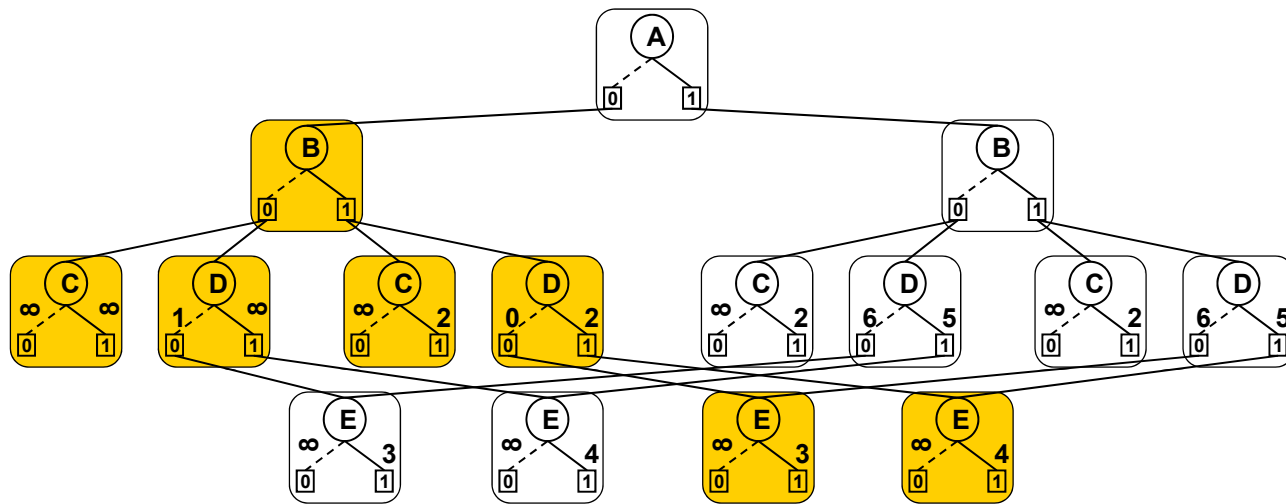
AOMDD – Compilation by Search



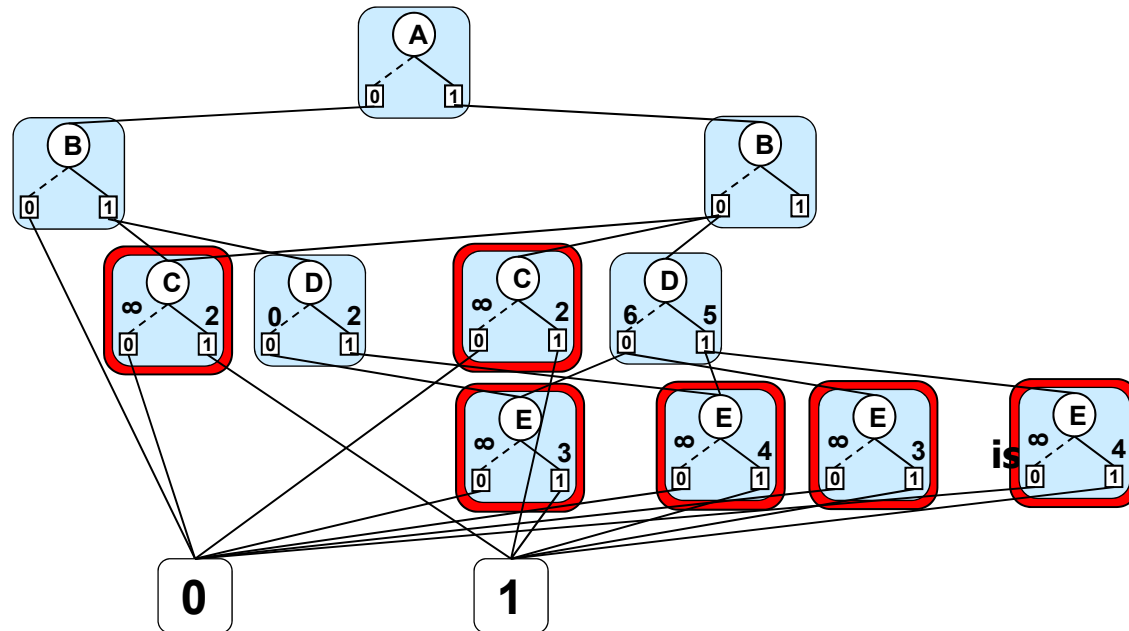
redundant



AOMDD – Compilation by Search

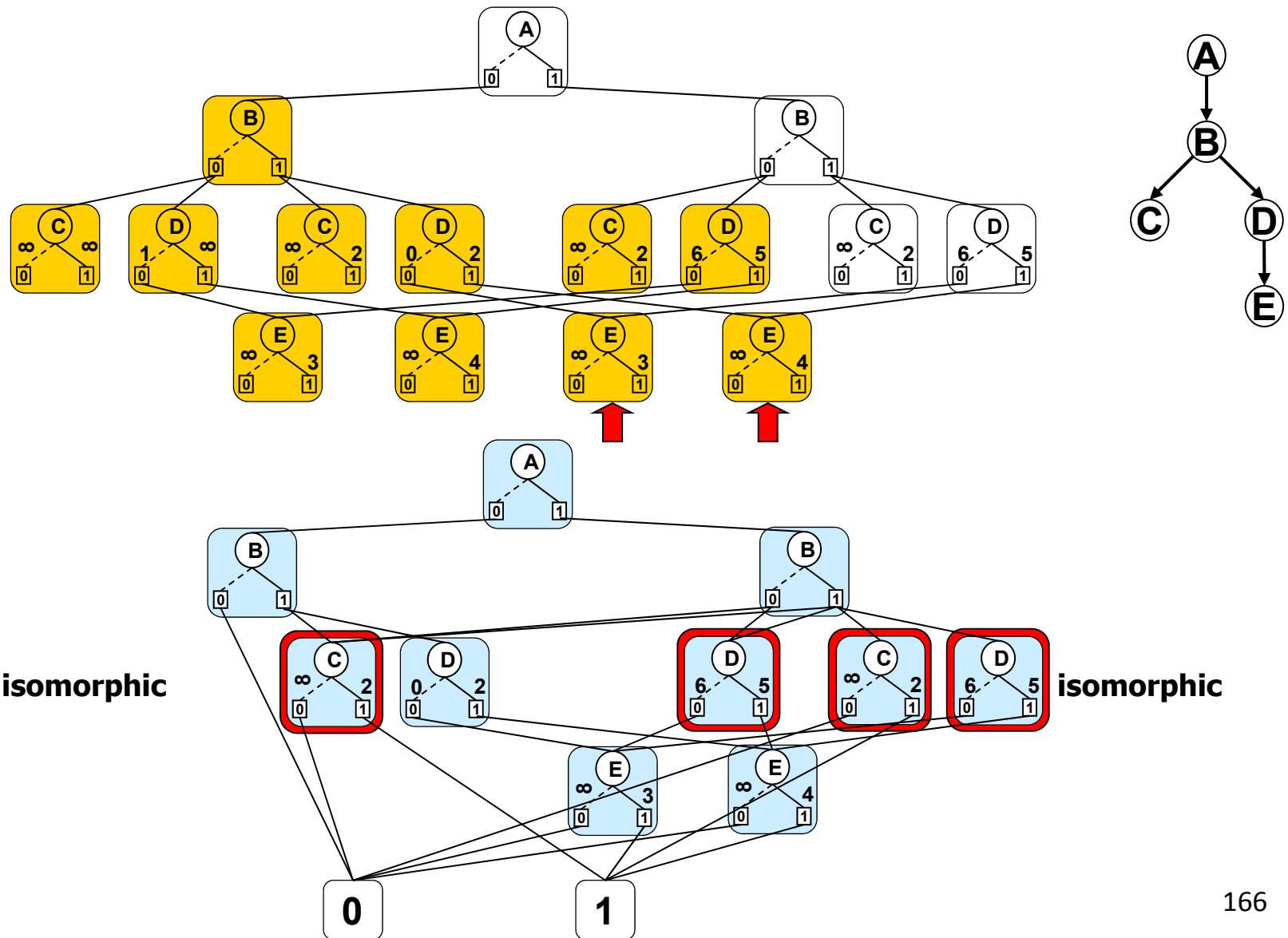


isomorphic

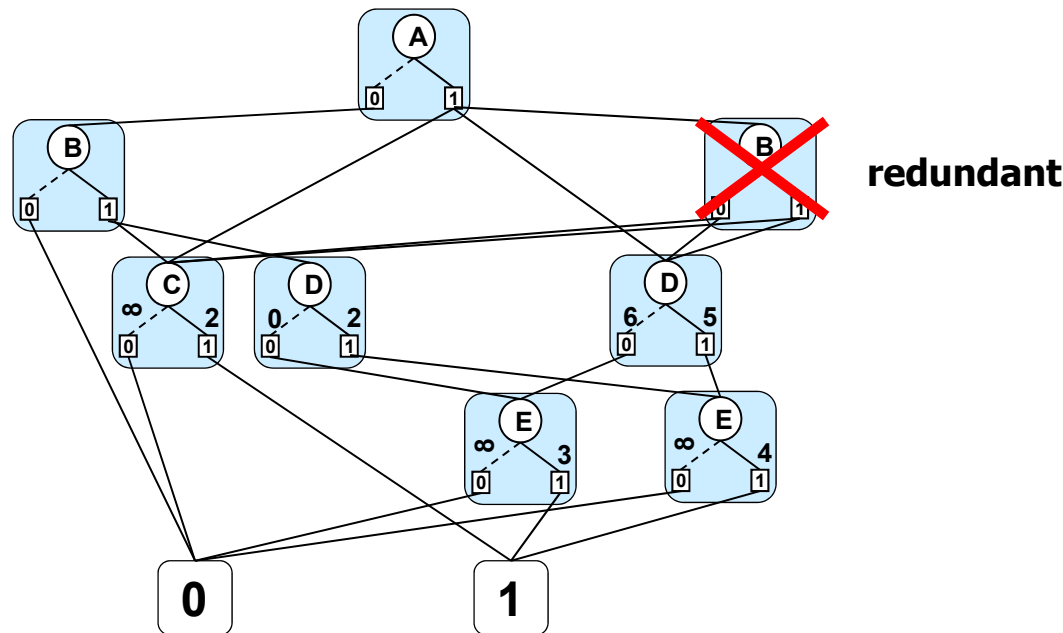
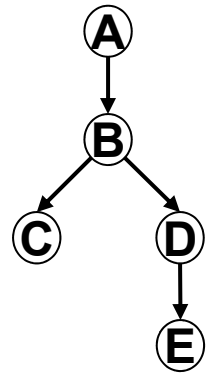
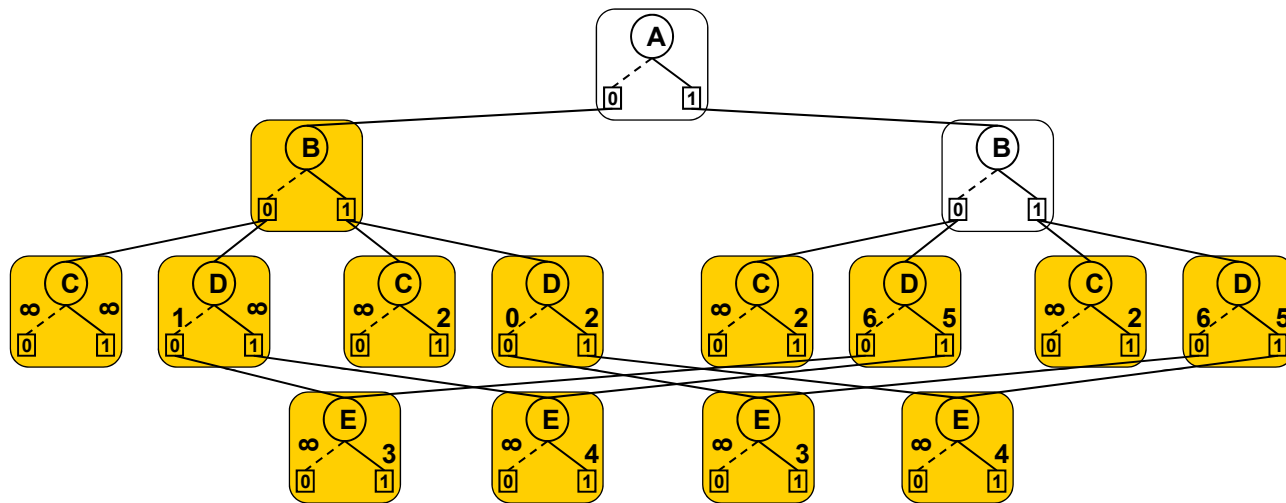


is isomorphic

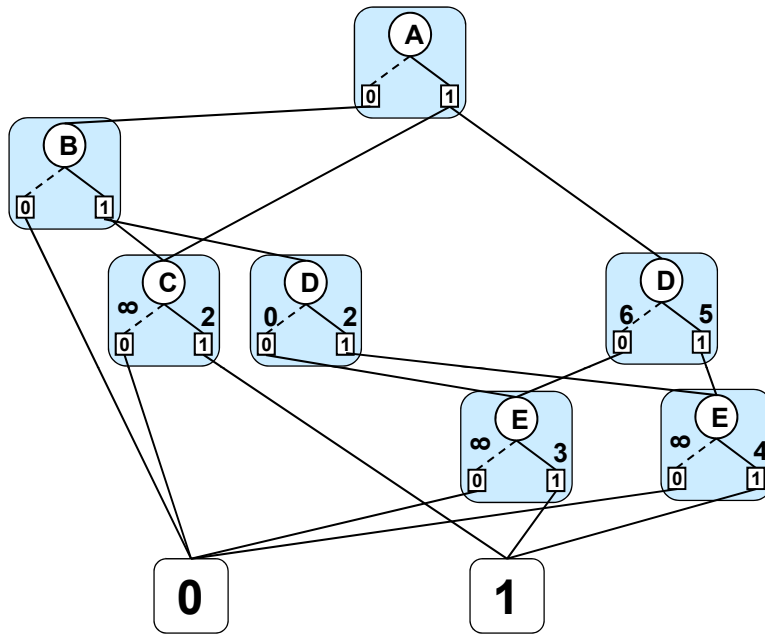
AOMDD – Compilation by Search



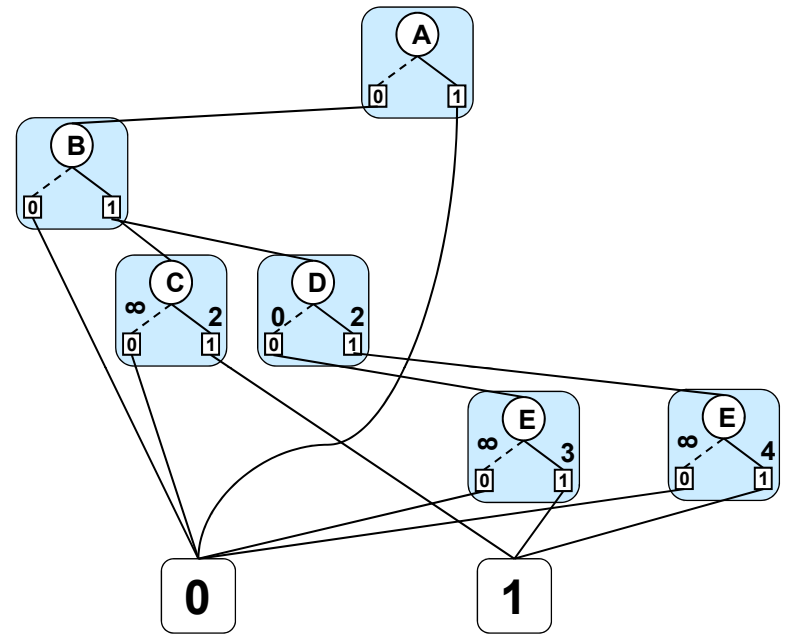
AOMDD – Compilation by Search



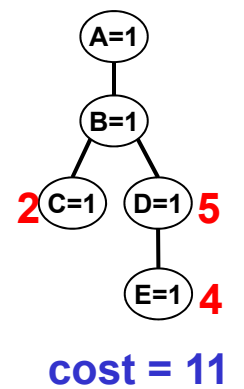
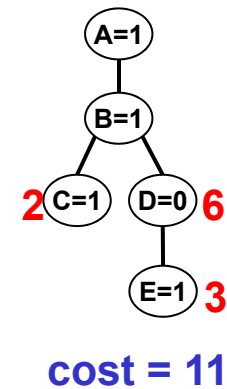
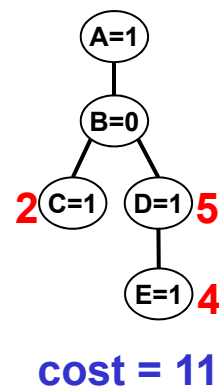
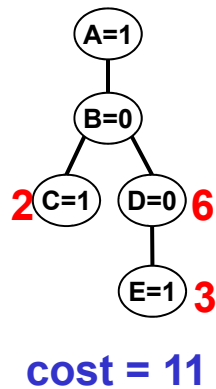
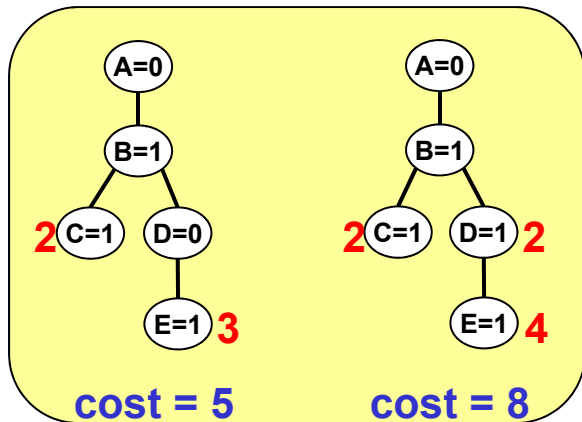
AOMDD for Constraint Optimization

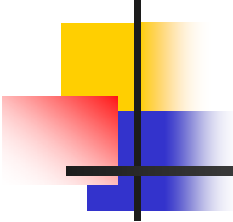


AOMDD for all solutions



AOMDD for two best solutions





Complexity of Compilation

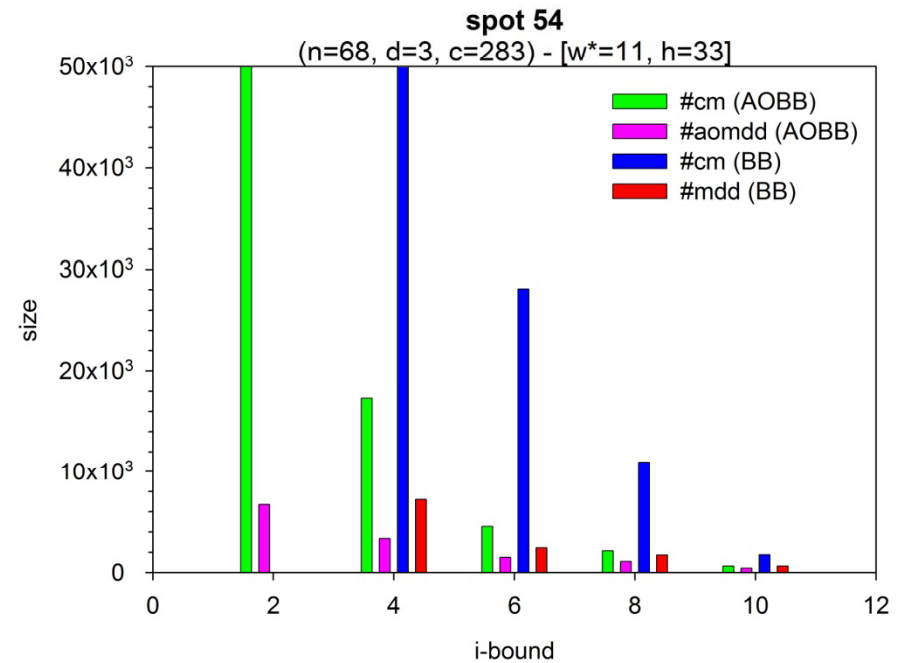
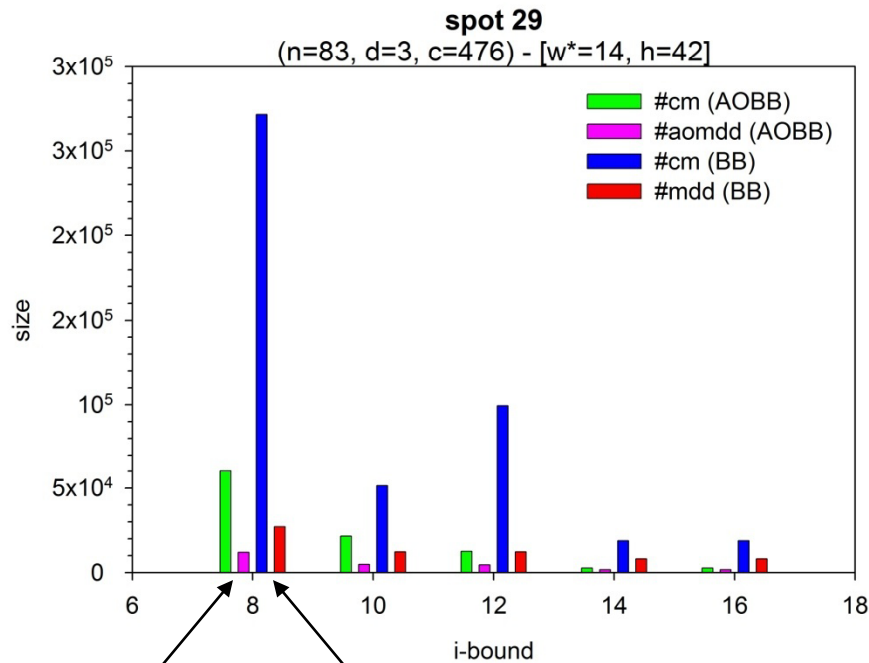
- The size of the AOMDD is $O(n k^{w^*})$
- The compilation time is also bounded by $O(n k^{w^*})$

k = domain size

n = number of variables

w^* = treewidth

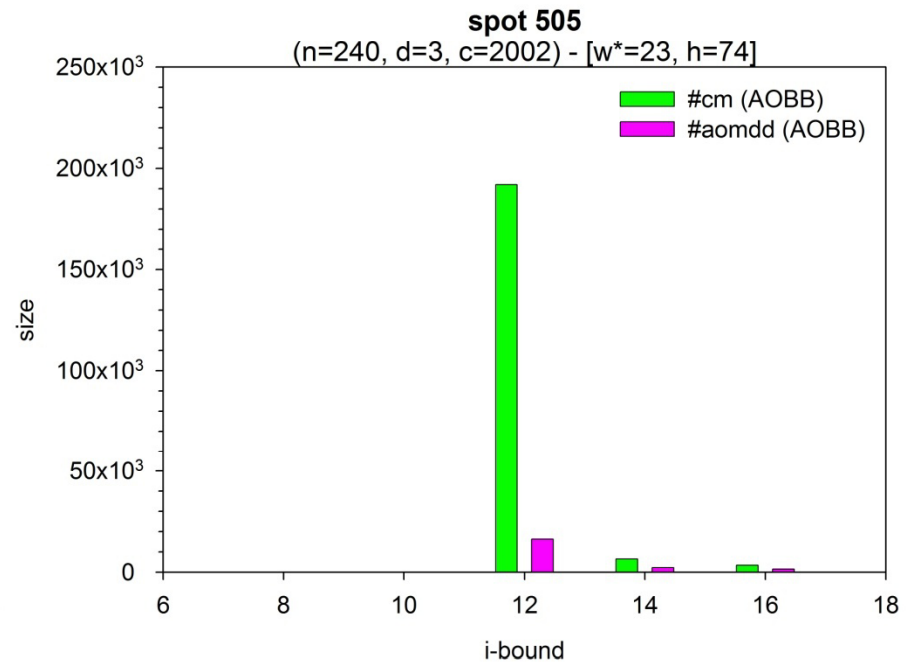
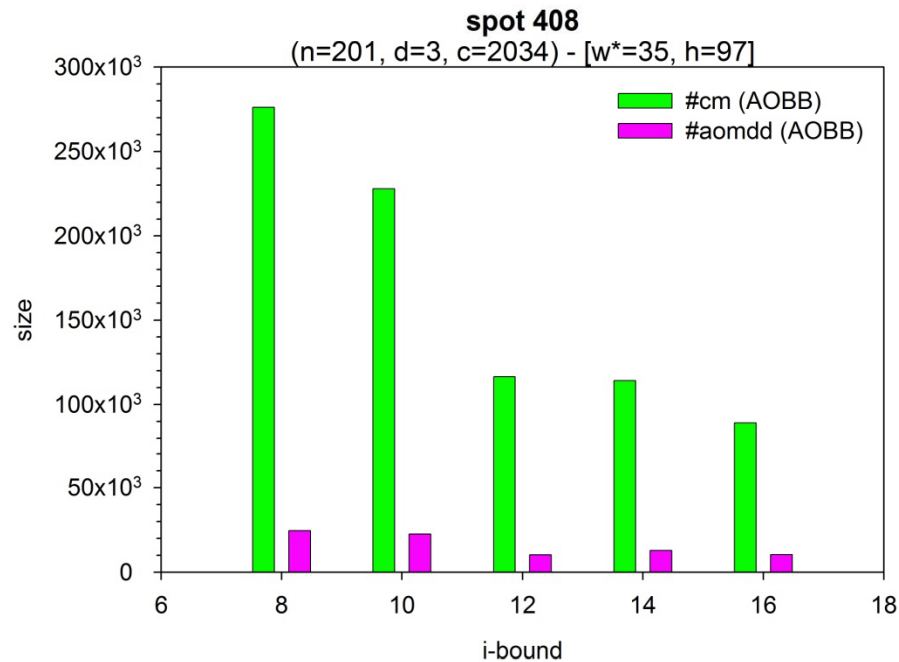
SPOT5 Benchmarks (WCSP)



AND/OR OR

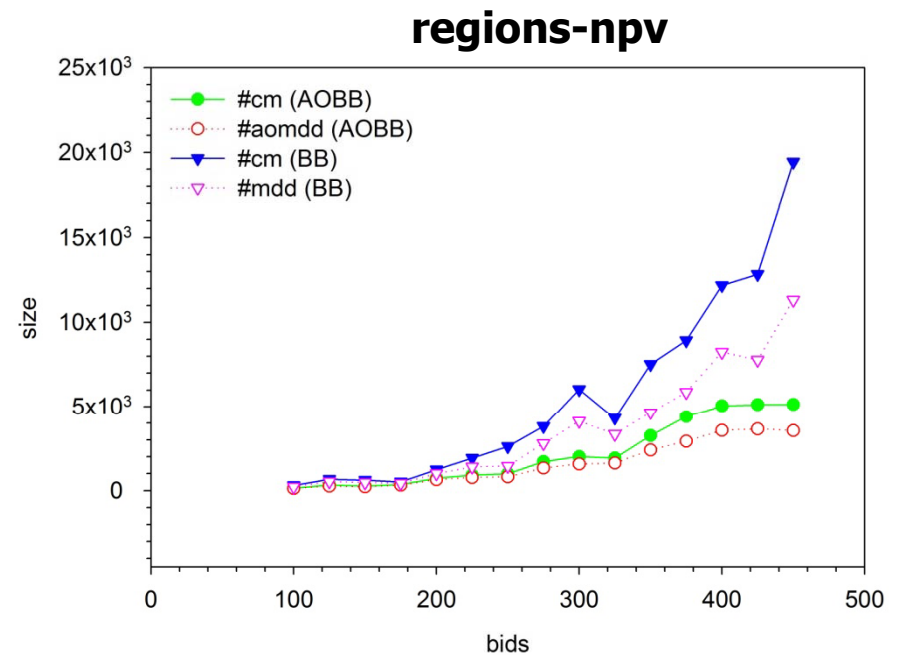
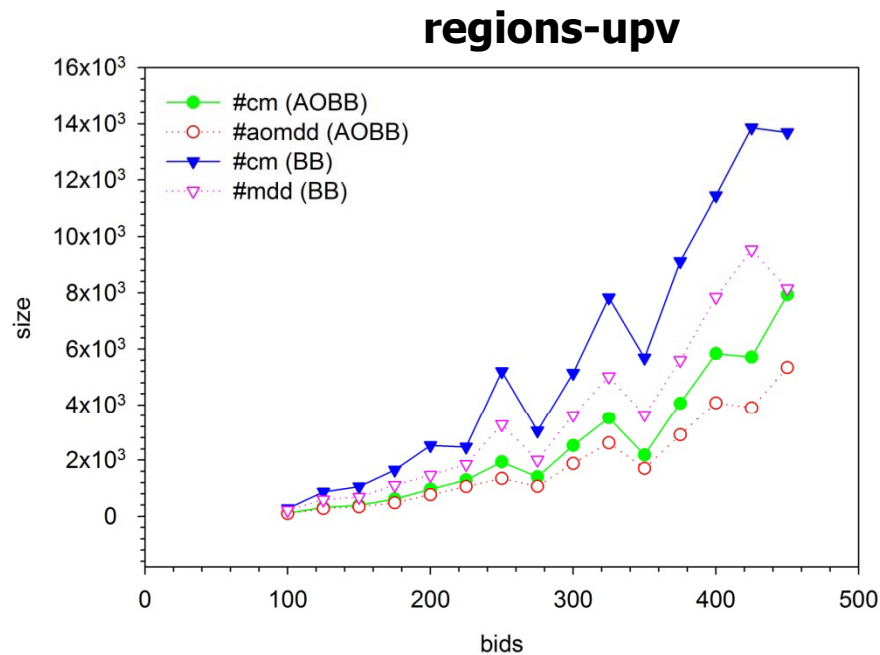
Results for Earth Observing Satellites benchmarks

SPOT5 Benchmarks (WCSP)



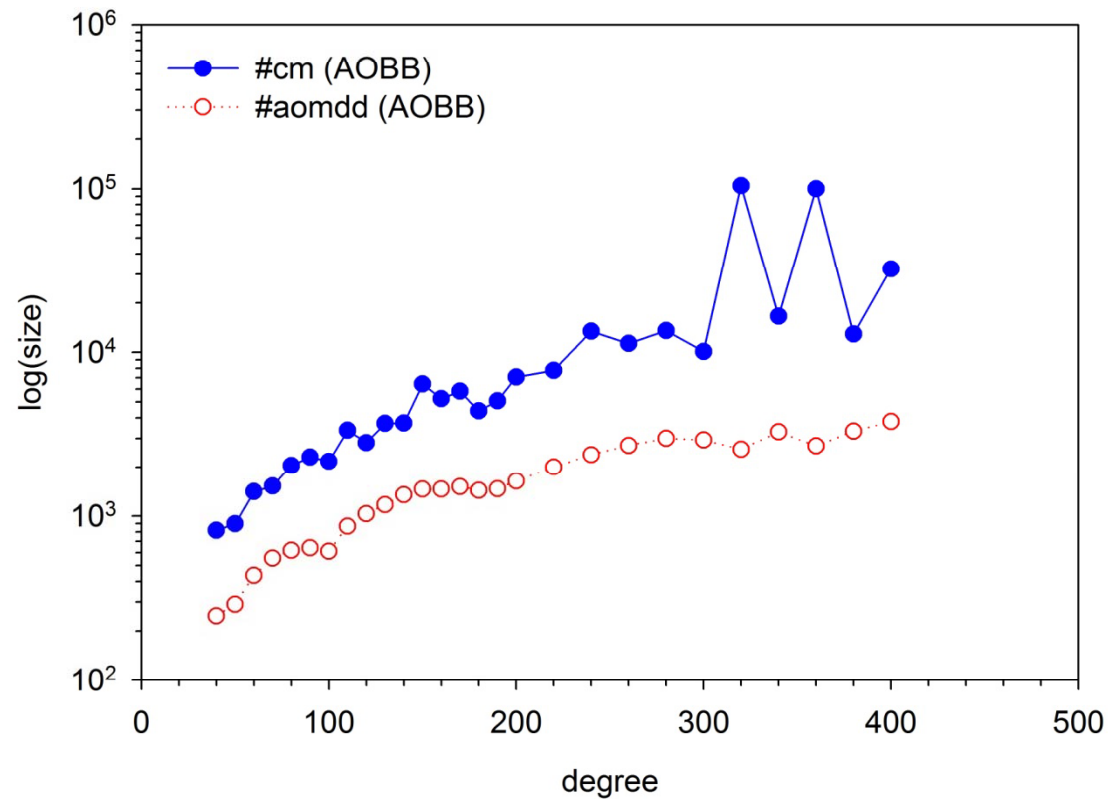
Results for Earth Observing Satellites benchmarks

Combinatorial Auctions (ILP)



Results for combinatorial auctions from the CATS 2.0 distribution

MAX-SAT Instances (ILP)



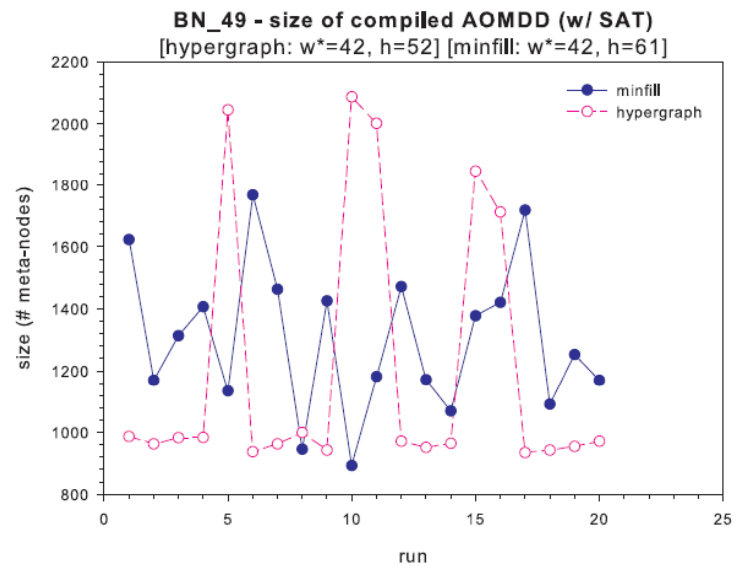
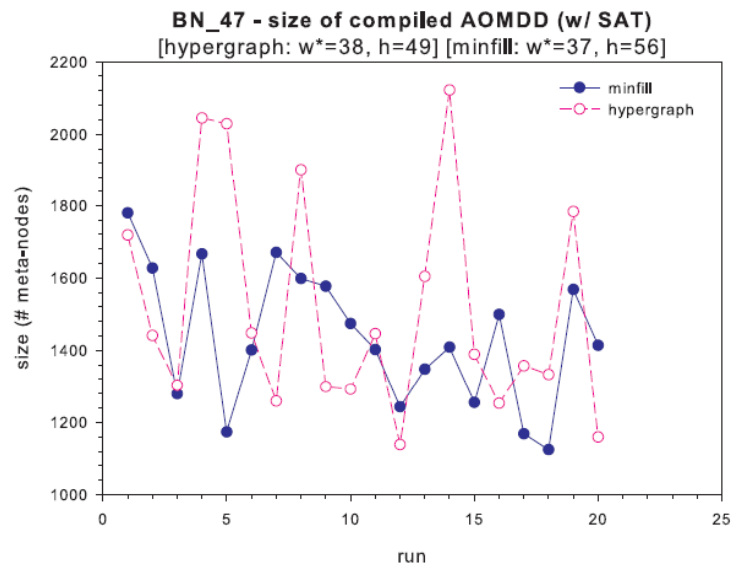
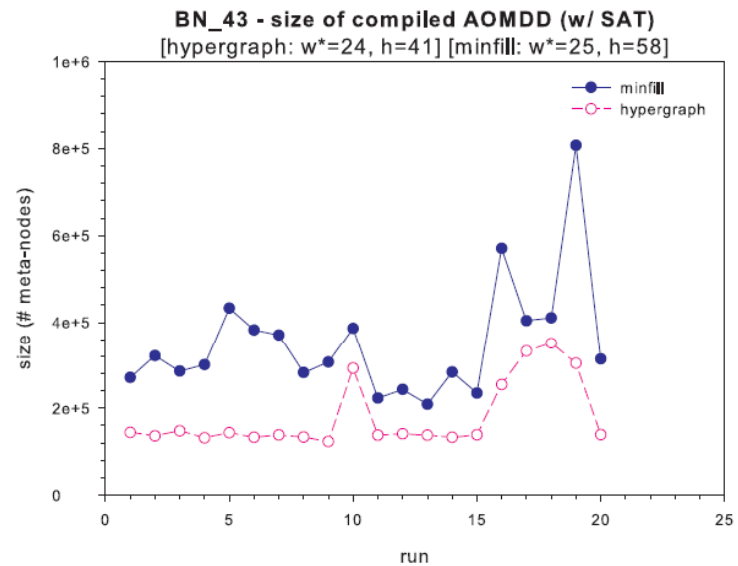
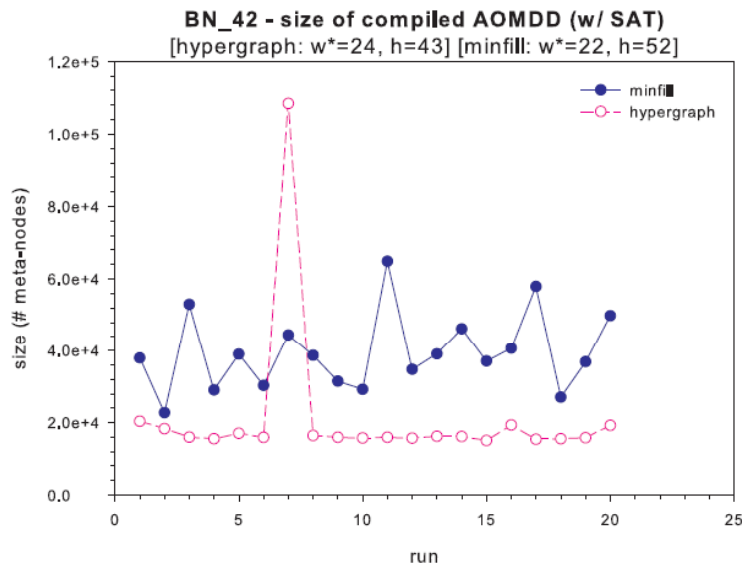
Results for dubois MAX-SAT instances

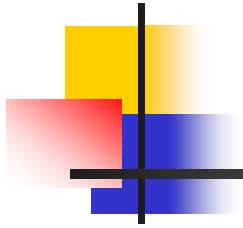
Bayesian Networks Repository

Network	(w*, h)	(n, k)	ACE		MDD w/ BCP			AOMDD w/ BCP			AOMDD w/ SAT		
			#nodes	time	#meta	#cm(OR)	time	#meta	#cm(OR)	time	#meta	#cm(OR)	time
Bayesian Network Repository													
alarm	(4, 13)	(37, 4)	1,511	0.01	208,837	682,195	73.35	320	459	0.05	320	459	0.22
cpcs54	(14, 23)	(54, 2)	196,933	0.06	-	-	-	65,158	66,405	6.97	65,158	66,405	6.97
cpcs179	(8, 14)	(179, 4)	67,919	0.05	-	-	-	9,990	32,185	46.56	9,990	32,185	46.56
cpcs360b	(20, 27)	(360, 2)	5,258,826	1.72	-	-	-	-	-	-	-	-	-
diabetes	(4, 77)	(413, 21)	7,615,989	1.81	-	-	-	-	-	-	-	-	-
hailfinder	(4, 16)	(56, 11)	8,815	0.01	-	-	-	2,068	2,202	0.34	1,893	2,202	1.48
mildew	(4, 13)	(35, 100)	823,913	0.39	-	-	-	73,666	110,284	1367.81	62,903	65,599	3776.82
mm	(20, 57)	(1220, 2)	47,171	1.49	-	-	-	38,414	58,144	4.54	30,274	52,523	99.55
munin2	(9, 32)	(1003, 21)	2,128,147	1.91	-	-	-	-	-	-	-	-	-
munin3	(9, 32)	(1041, 21)	1,226,635	1.27	-	-	-	-	-	-	-	-	-
munin4	(9, 32)	(1044, 21)	2,423,009	4.44	-	-	-	-	-	-	-	-	-
pathfinder	(6, 11)	(109, 63)	18,250	0.05	610,854	1,303,682	352.18	6,984	16,267	30.71	2,265	15,963	50.36
pigs	(11, 26)	(441, 3)	636,684	0.19	-	-	-	261,920	294,101	174.29	198,284	294,101	1277.72
water	(10, 15)	(32, 4)	59,642	0.52	707,283	1,138,096	95.14	18,744	20,926	2.02	18,503	19,225	7.45

Size (number of nodes), time (seconds)

Effect of Variable Ordering

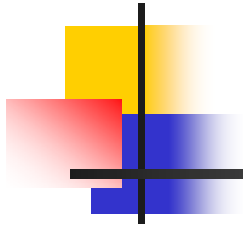




Outline

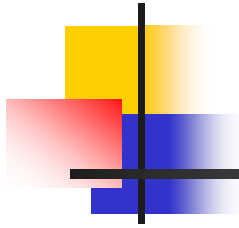
- Introduction
- Inference
- Search (OR)
- Lower-bounds and relaxations
- Exploiting problem structure in search

- **Software**
 - aolib and toulbar2 software packages
 - Results from UAI-2006, CP-2006, UAI-2008 and UAI-2010 solver competitions



Outline

- Introduction
- Inference
- Search
- Compilation: AND/OR Decision Diagrams
- **Software**



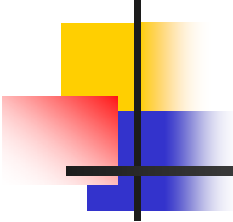
Software

■ Reports on competitions

- UAI'06 Inference Evaluation
 - 57 MPE instances
- CP'06 Competition
 - 686 2-ary MAX-CSP instances
 - 135 n-ary MAX-CSP instances
- UAI'08 Competition
 - 480 MPE instances

■ How to use the software

- <http://graphmod.ics.uci.edu/group/Software>
- <http://mulcyber.toulouse.inra.fr/projects/toulbar2>



Toulbar2 and aolib

- toulbar2

<http://mulcyber.toulouse.inra.fr/gf/project/toulbar2>

(Open source WCSP, MPE solver in C++)

- aolib

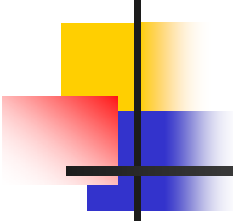
<http://graphmod.ics.uci.edu/group/Software>

(WCSP, MPE, ILP solver in C++, inference and counting)

- Large set of benchmarks

<http://carlit.toulouse.inra.fr/cgi-bin/awki.cgi/SoftCSP>

<http://graphmod.ics.uci.edu/group/Repository>



UAI-2006 Competition

- **Team 1 (UCLA)**

- David Allen, Mark Chavira, Arthur Choi, Adnan Darwiche

- **Team 2 (IET)**

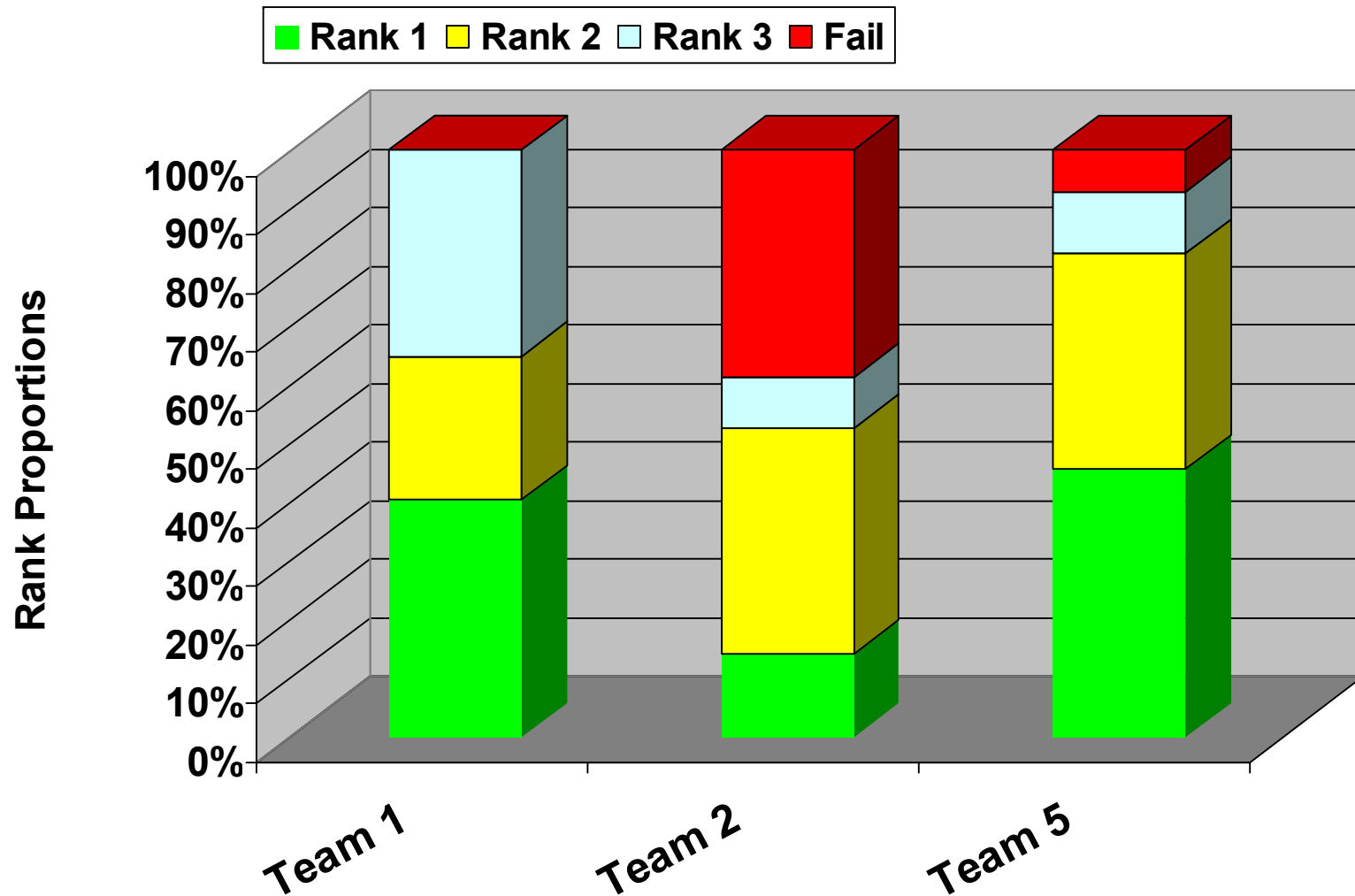
- Masami Takikawa, Hans Dettmar, Francis Fung, Rick Kissh

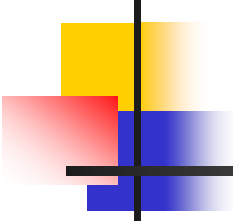
- **Team 5 (UCI)**

- Radu Marinescu, Robert Mateescu, Rina Dechter
- Used **AOBB-C+SMB(i)** solver for MPE

UAI-2006 Results

Rank Proportions (how often was each team a particular rank, rank 1 is best)

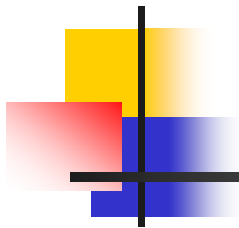




CP-2006 Competition (WCSP)

■ Solvers

- AbsconMax (ie, DFBB+MRDAC)
- **aolibdvo** (ie, AOBB+EDAC+DVO solver)
- **aolibpvo** (ie, AOBB+EDAC+PVO solver)
- CSP4J-MaxCSP
- **Toolbar** (ie, DFBB+EDAC)
- **Toolbar_BT D** (ie, BT D+EDAC+VE)
- **Toolbar_MaxSAT** (ie, DPLL+specific EPT rules)
- **Toulbar2** (ie, DFBB+EDAC+VE+LDS)

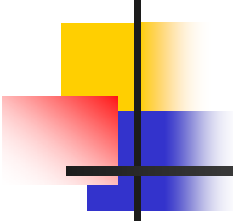


CP-2006 Results

Overall ranking on all selected competition benchmarks

Solver Name		Progress				
AbsconMax 109 EPFC	done 1069					
	MOPT 479	SAT 26	MSAT 563		Inc. Answer 1	
AbsconMax 109 PFC	done 1069					
	MOPT 500	SAT 26	MSAT 542		Inc. Answer 1	
aolibdvo 2007-01-17	done 821					
	MOPT 495	SAT 25	MSAT 42	? 259		
aolibpvo 2007-01-17	done 821					
	MOPT 490	SAT 25	MSAT 47	? 258 ERR 1		
CSP4J - MaxCSP 2006-12-19	done 1069					
	MOPT 2	SAT 26	MSAT 592		? 449	
toolbar 2007-01-12	done 821					
	MOPT 641	SAT 26	MSAT 93	? 61		
Toolbar_BT 2007-01-12	done 821					
	MOPT 646	SAT 26	? 149			
Toolbar_MaxSat 2007-01-19	done 821					
	MOPT 202	SAT 26	? 587 ERR 6			
Toulbar2 2007-01-12	done 821					
	MOPT 593	SAT 26	MSAT 151		? 51	

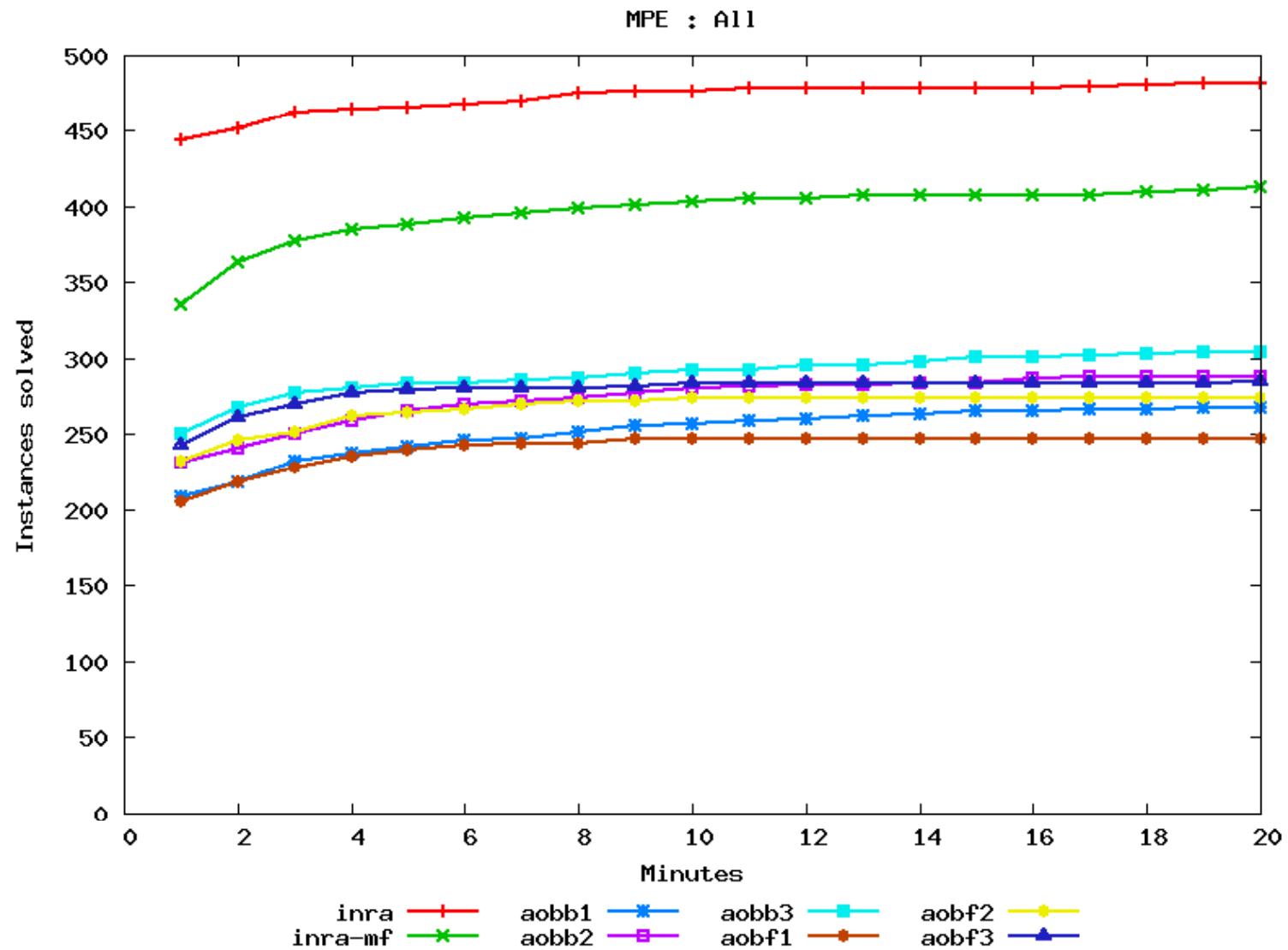
The longest **dark green** bar wins

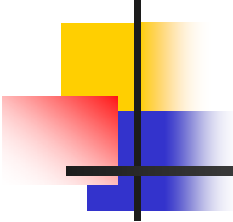


UAI-2008 Competition

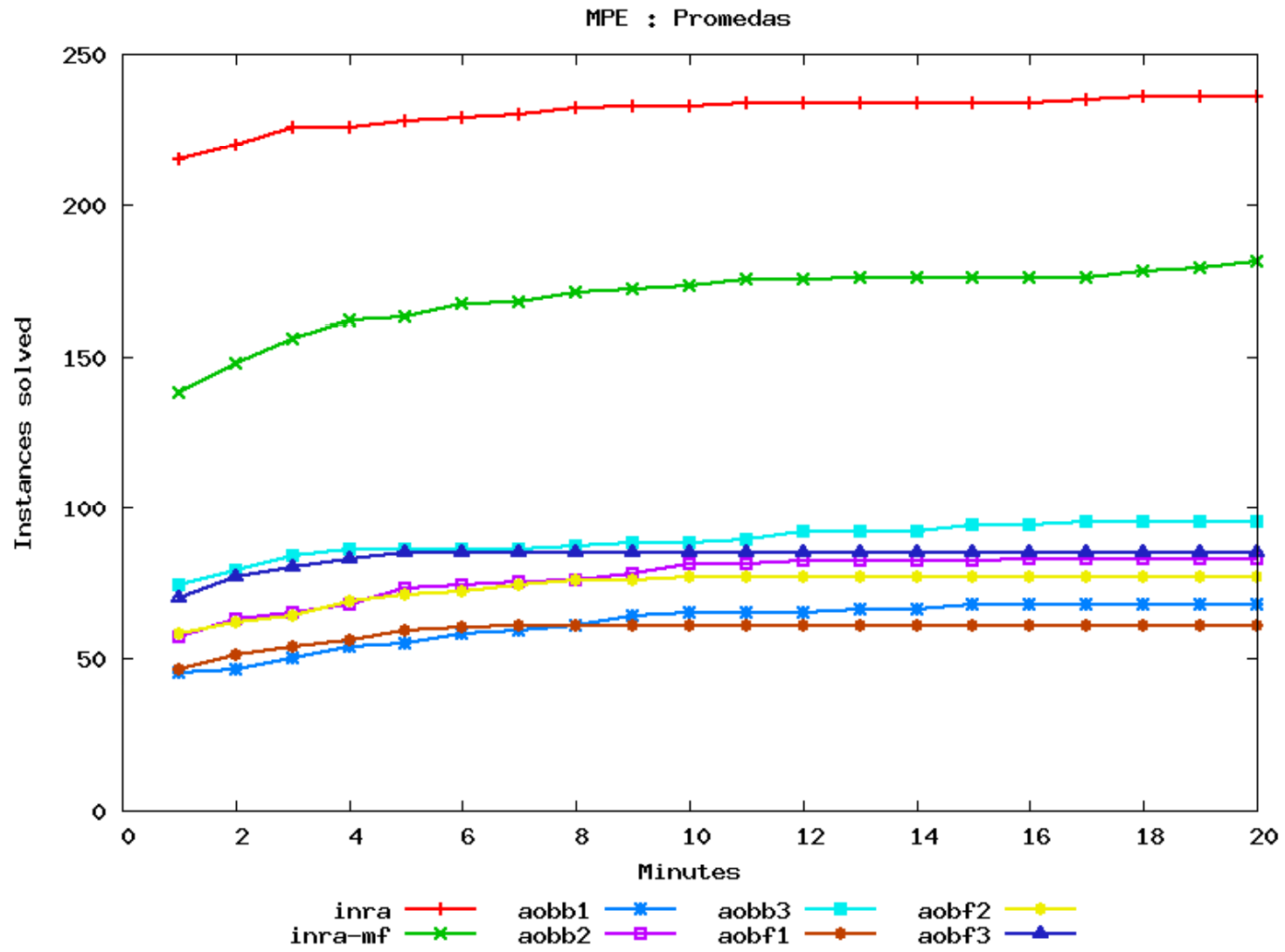
- **AOBB-C+SMB(i) – (i = 18, 20, 22)**
 - AND/OR Branch-and-Bound with pre-compiled mini-bucket heuristics (i-bound), full caching, static pseudo-trees, constraint propagation
- **AOBF-C+SMB(i) – (i = 18, 20, 22)**
 - AND/OR Best-First search with pre-compiled mini-bucket heuristics (i-bound), full caching, static pseudo-trees, no constraint propagation
- **Toulbar2**
 - OR Branch-and-Bound, dynamic variable/value orderings, EDAC consistency for binary and ternary cost functions, variable elimination of small degree (2) during search
- **Toulbar2/BTD**
 - DFBB exploiting a tree decomposition (AND/OR), same search inside clusters as toulbar2, full caching (no cluster merging), combines RDS and EDAC, and caching lower bounds

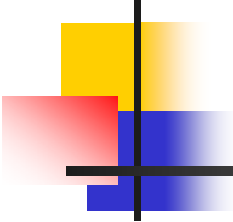
UAI-2008 Results





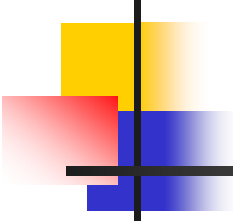
UAI-2008 Results (contd.)





UAI-2010 Competition

- Tasks
 - PR: probability of evidence
 - MAR: posterior marginals
 - MPE: most probable explanation
- 3 tracks: 20 sec, 20 min, 1 hour
 - PR, MAR - 204 instances; MPE - 442 instances
 - CSP, grids, image alignment, medical diagnosis, object detection, pedigree, protein folding, protein-protein interaction, relational model, segmentation
- Exact and approximate solvers



UAI-2010 Results

- MAR task
 - **1st place** – 20 min and 1 hour tracks
 - Anytime IJGP(i) with randomized orderings and SAT based domain pruning (Mateescu et al, JAIR2010), (Dechter et al, UAI2002)
- PR task
 - **1st place** – 20 min and 1 hour tracks
 - Formula SampleSearch with IJGP(3) based importance distribution, w-cutset sampling, minisat based search, rejection control (Gogate, Domingos and Dechter UAI2010)
- MPE task
 - **3rd place** – all tracks
 - AND/OR Branch and Bound with mini-buckets (Marinescu and Dechter, AIJ2009), (Otten and Dechter, ISAIM2010)